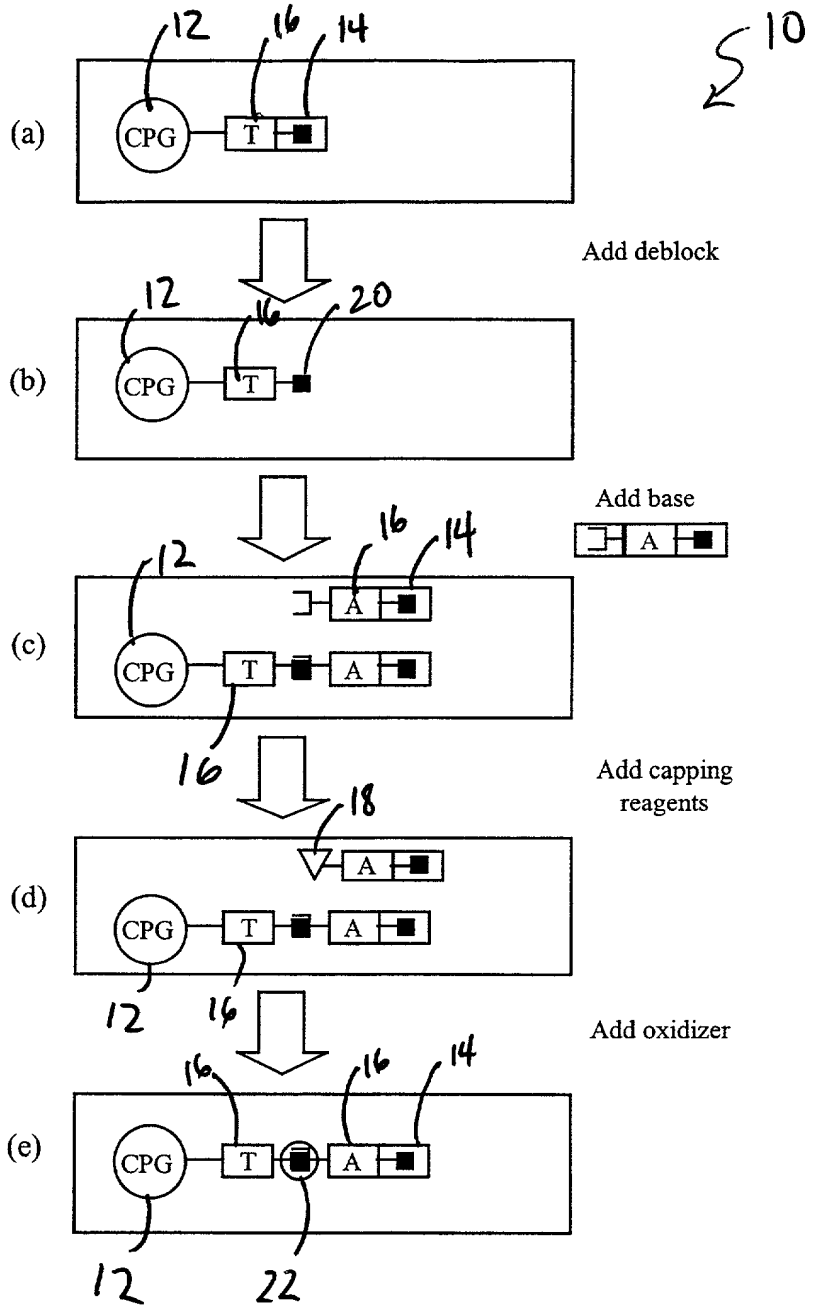

10
(a)
12
16
14
CPG
T
Add deblock
(b)
12
16
20
CPG
T
Add base
A
(c)
12
16
14
CPG
T
A
16
Add capping reagents
(d)
18
A
CPG
T
A
12
16
Add oxidizer
(e)
16
16
14
CPG
T
A
12
22

Figure 1

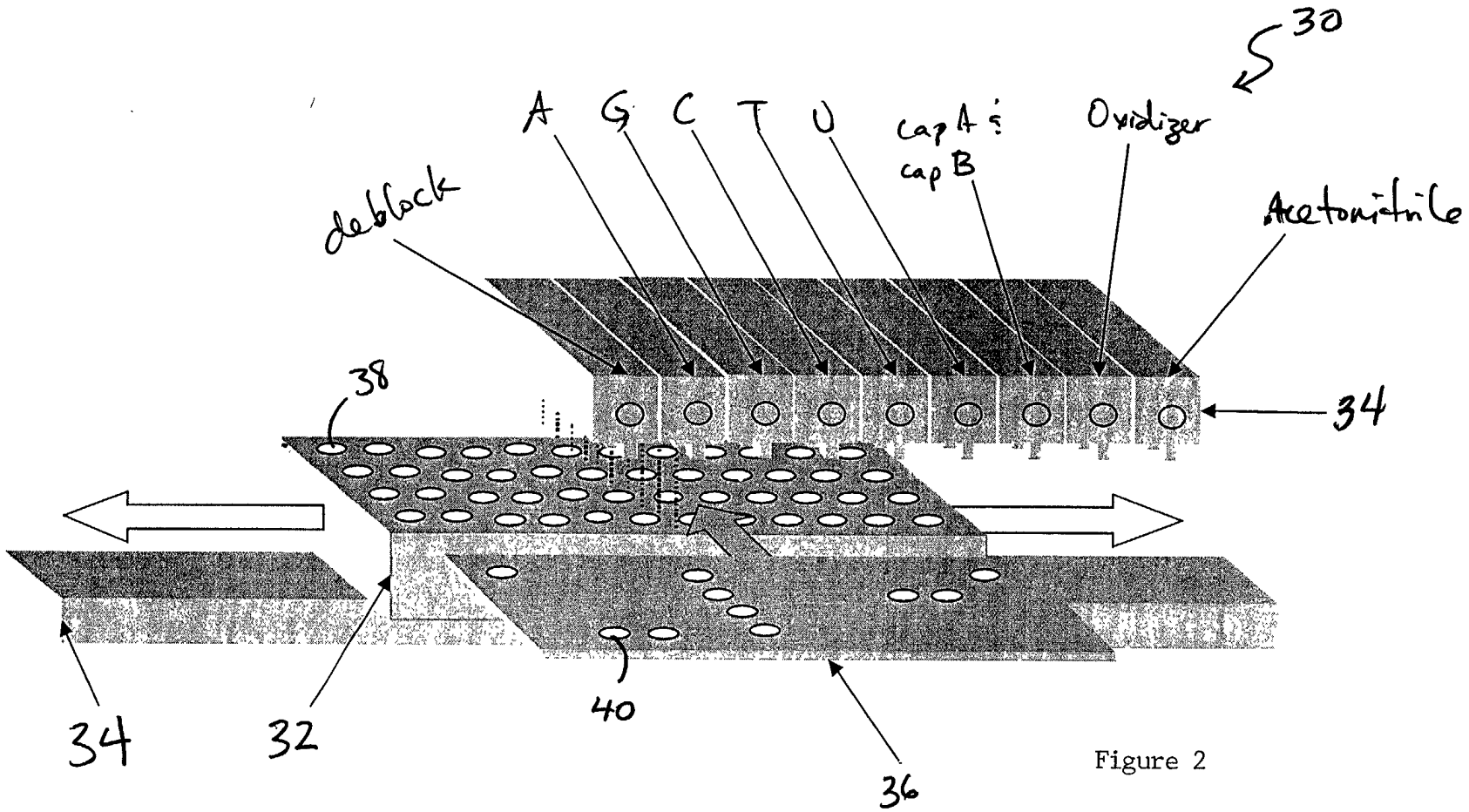

30
A
G
C
T
U
cap A &
cap B
Oxidizer
deblock
Acetonitrile
38
34
34
32
40
36

Figure 2

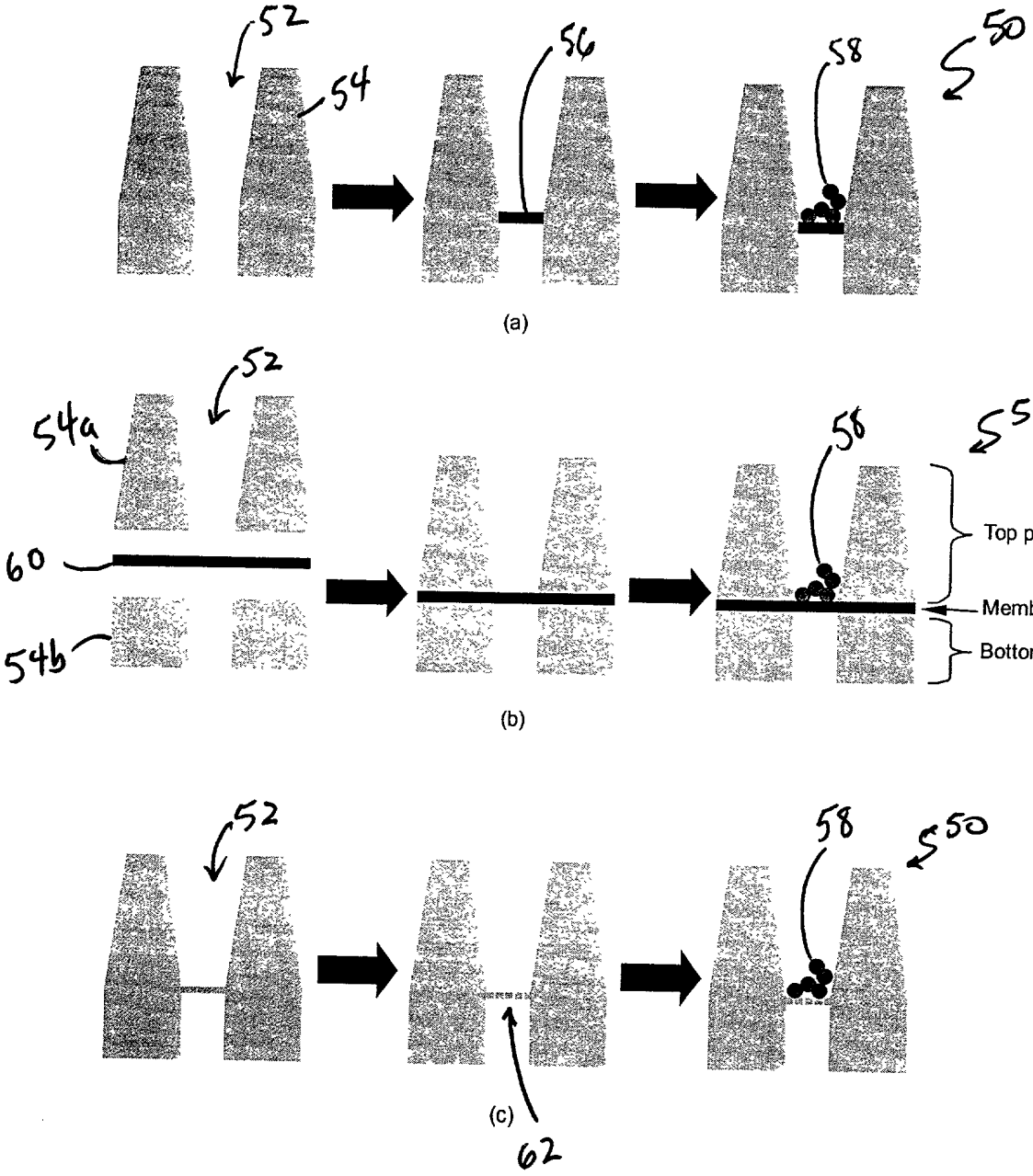
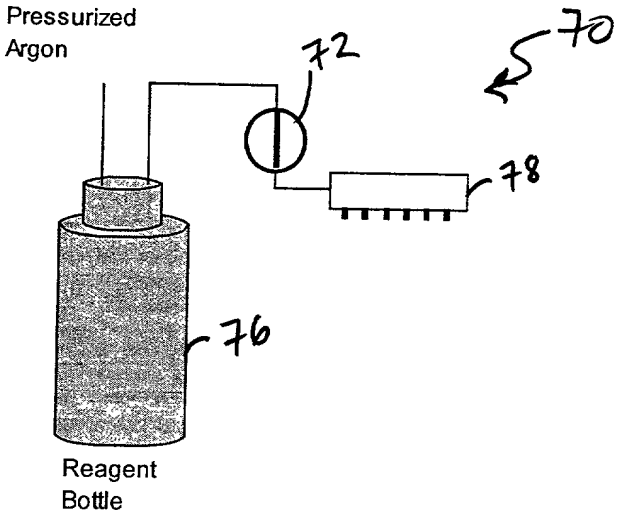

52
54
56
58
50
(a)
52
54a
60
54b
56
50
Top plate
Membrane
Bottom plate
(b)
52
58
50
(c)
62

Figure 3

(a)


Pressurized
Argon
72
70
78
76
Reagent
Bottle

(b)

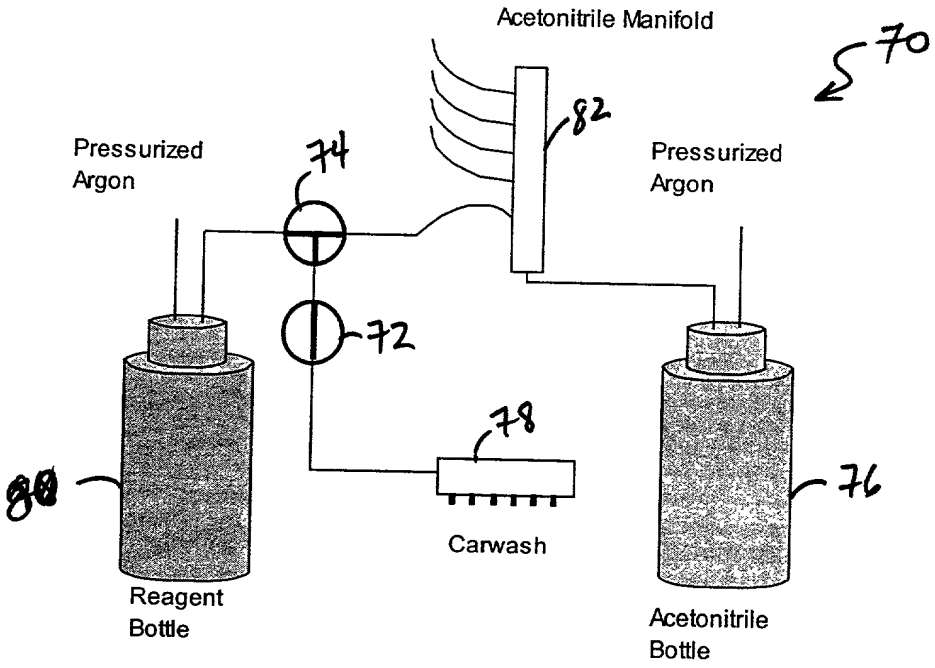

Acetonitrile Manifold
70
82
Pressurized
Argon
74
Pressurized
Argon
72
78
80
76
Carwash
Reagent
Bottle
Acetonitrile
Bottle

Figure 4

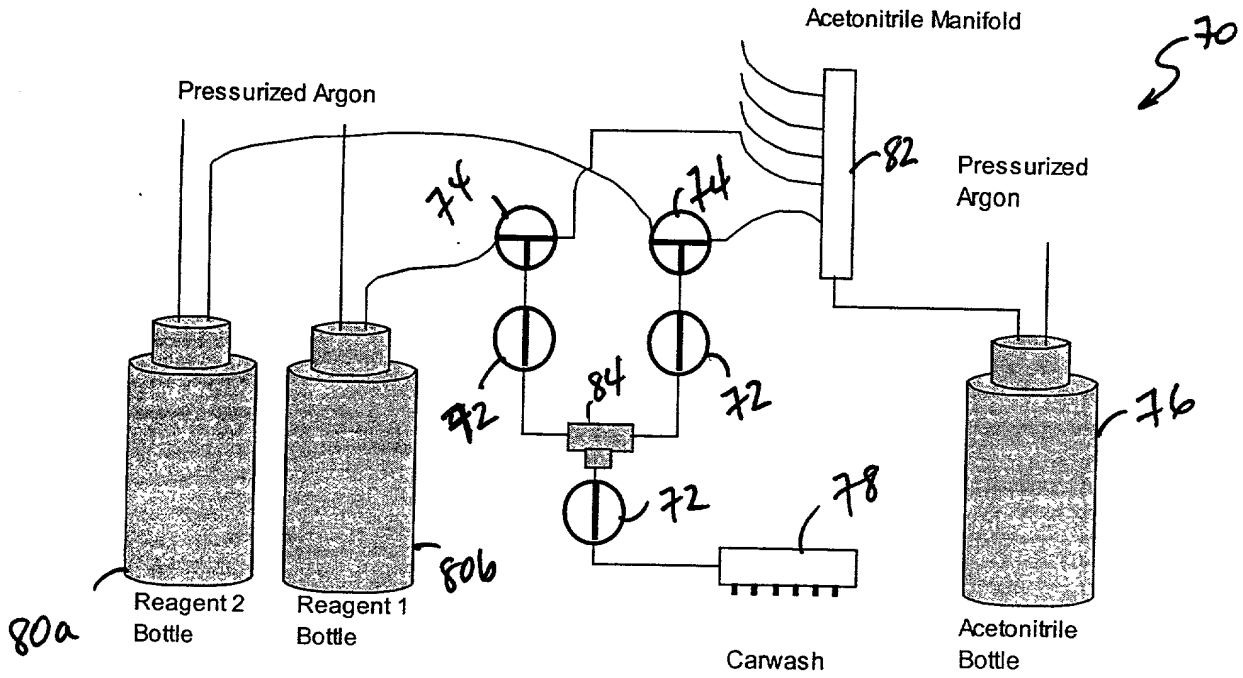


Figure 5a

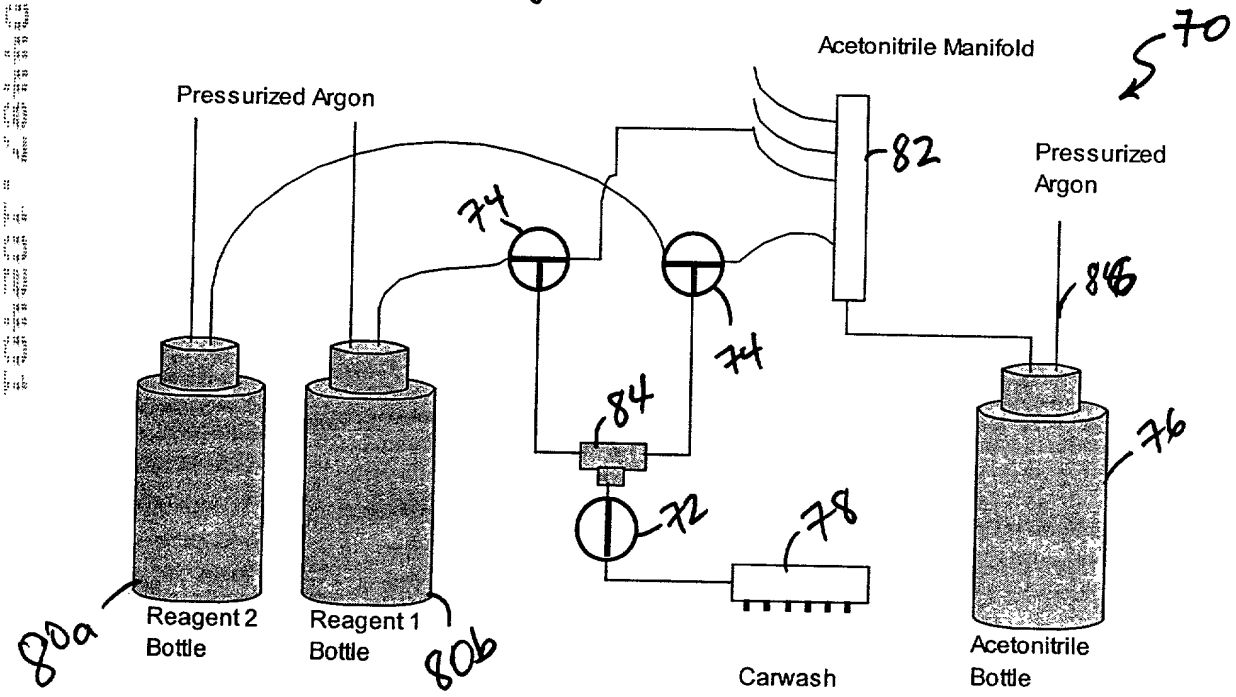


Figure 5b

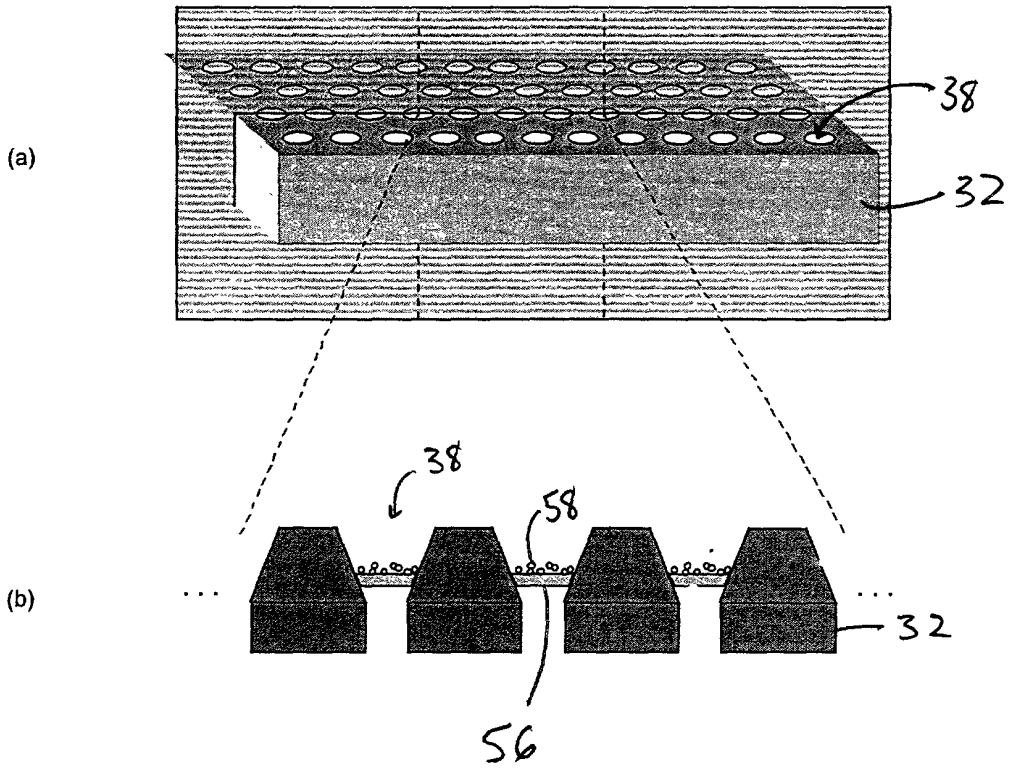


Figure 6

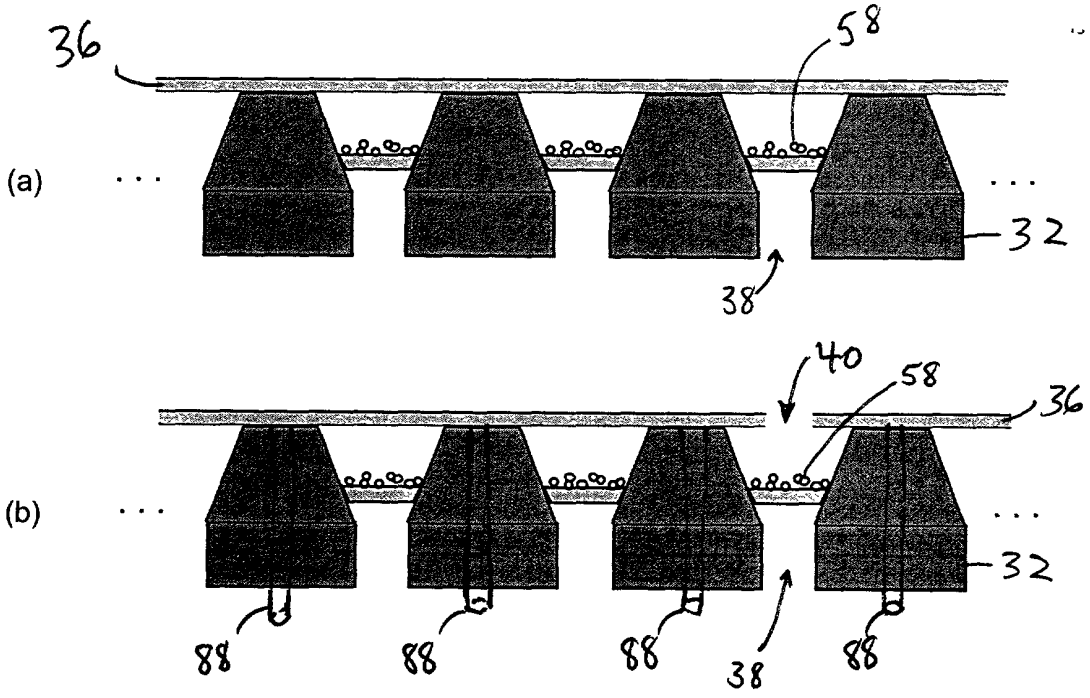


Figure 7

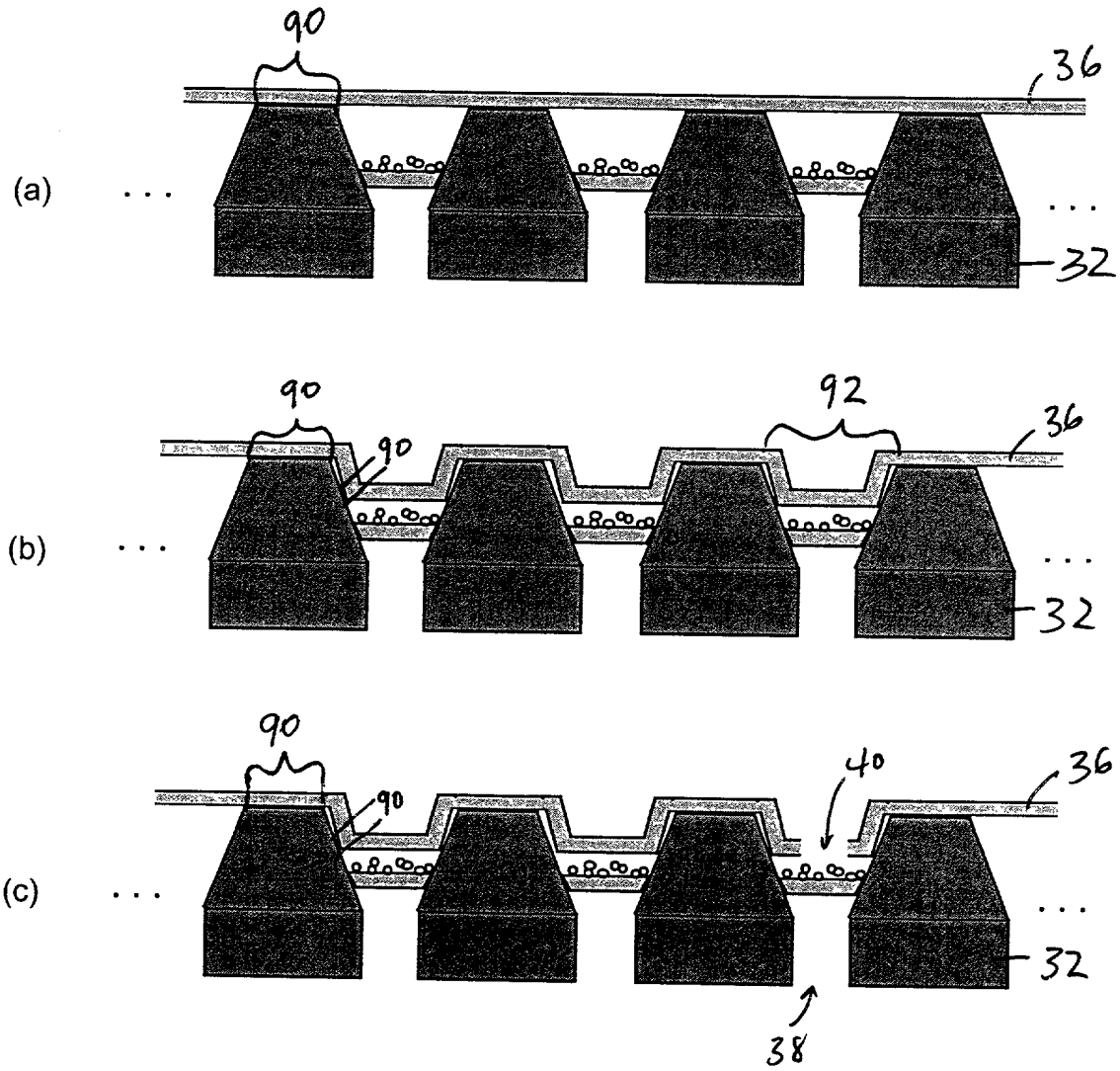


Figure 8

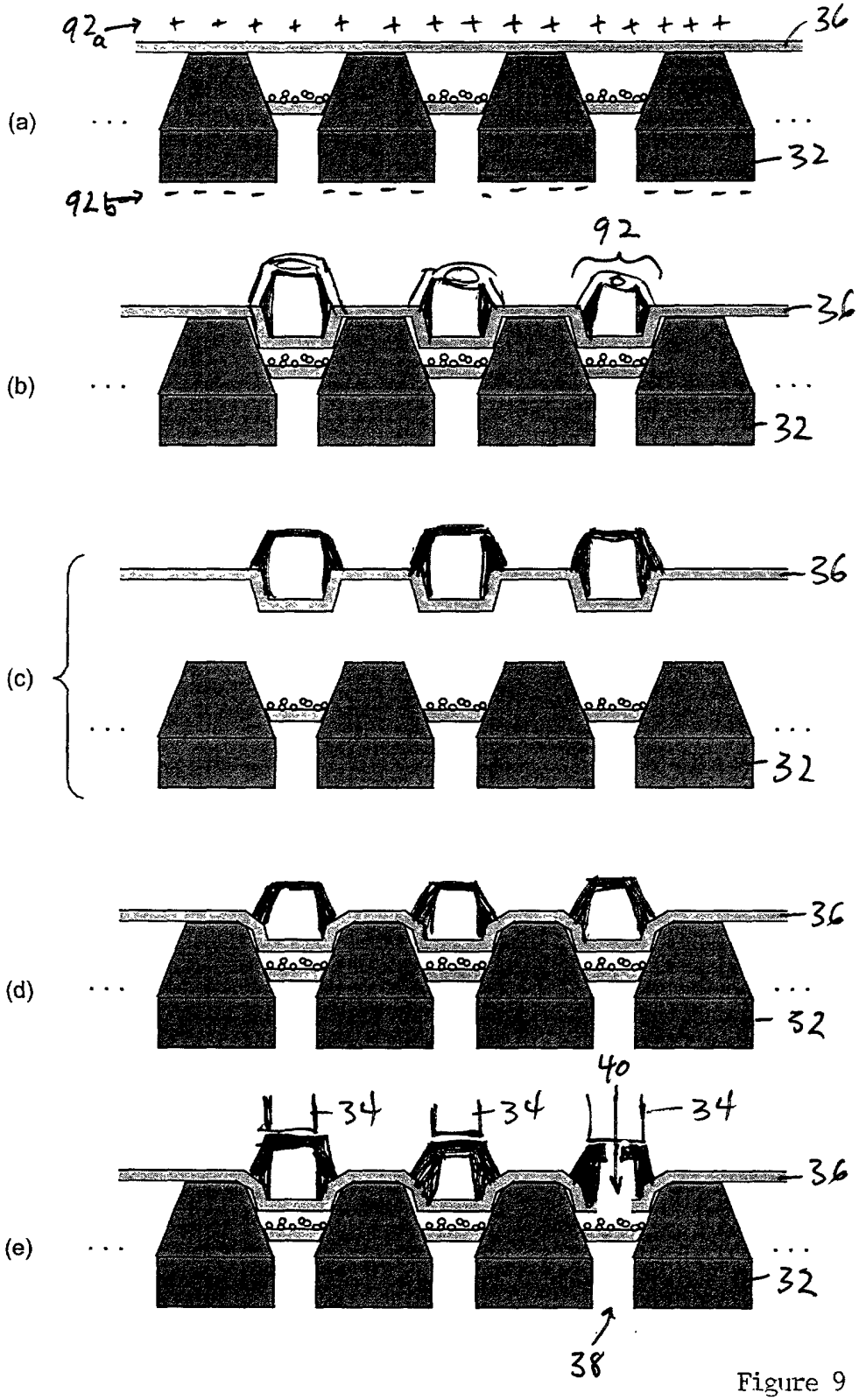


Figure 9

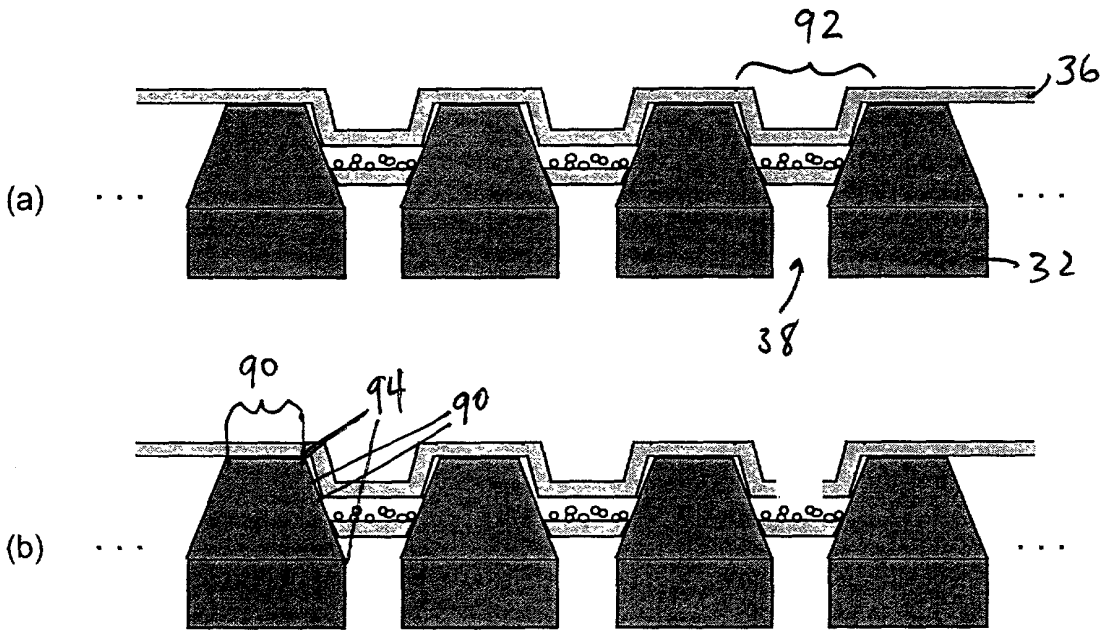


Figure 10

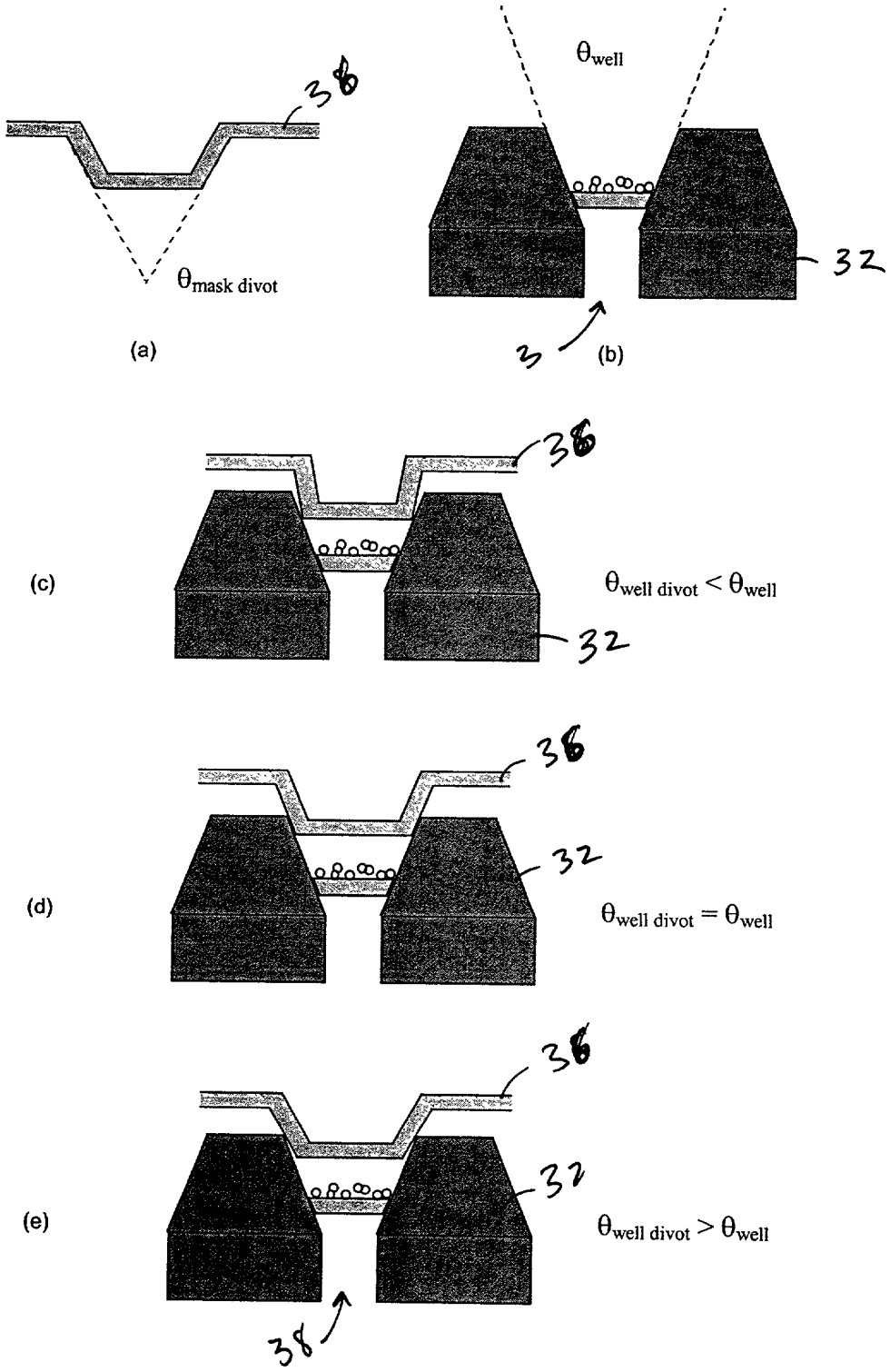

θwell
38
θmask divot
32
(a)
(b)
(c)
θwell divot < θwell
(d)
θwell divot = θwell
(e)
θwell divot > θwell

Figure 11

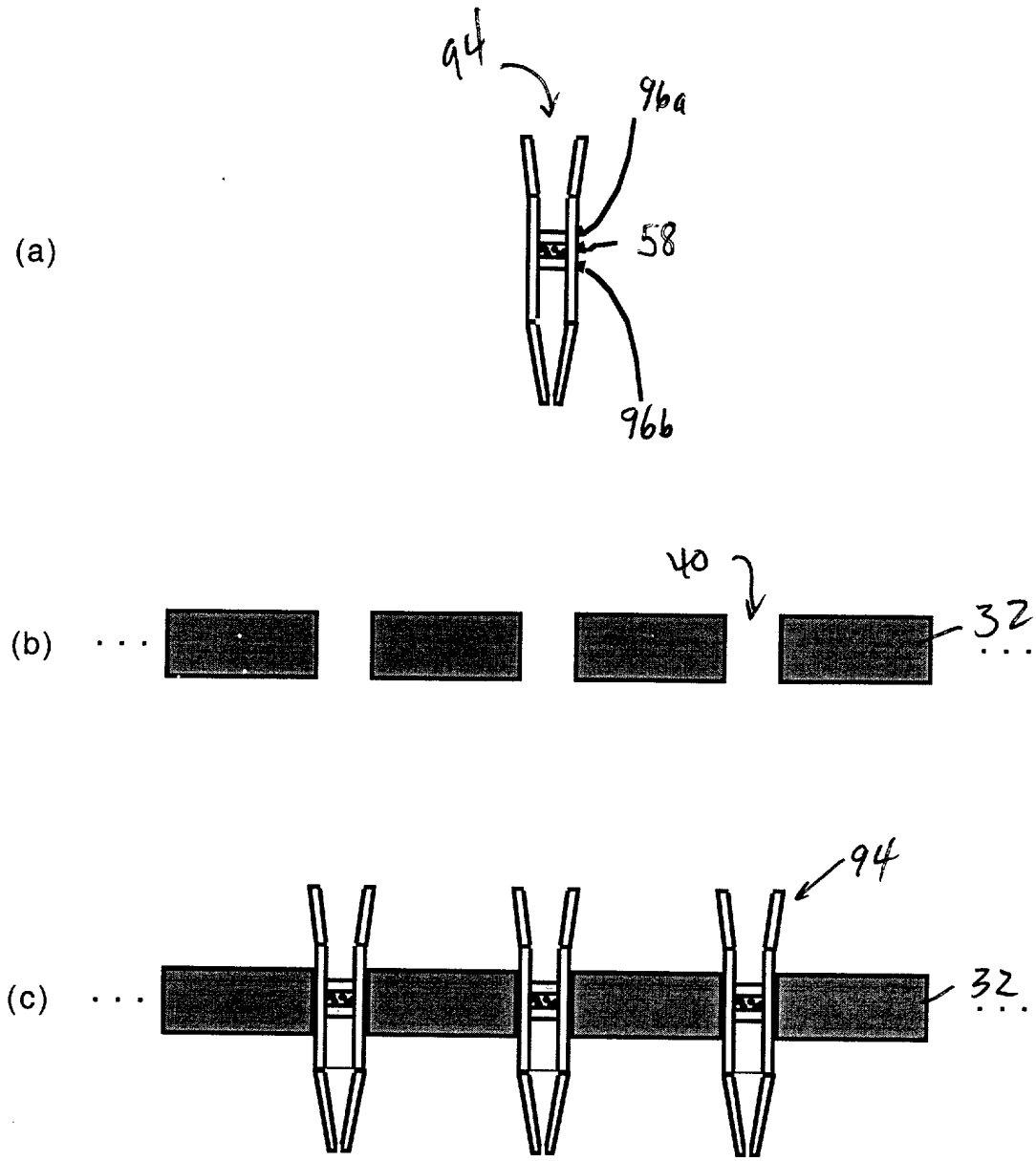


Figure 12

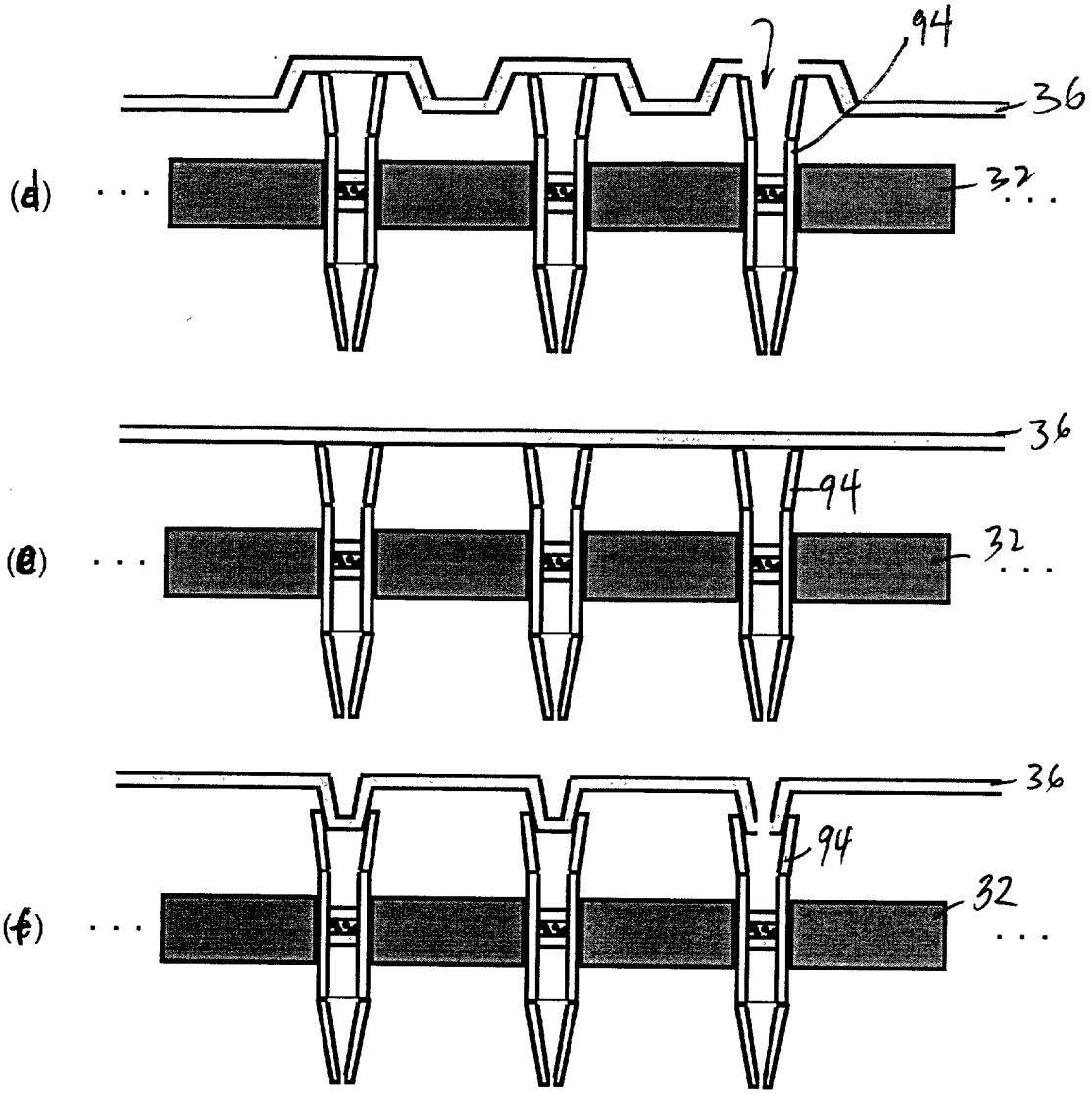
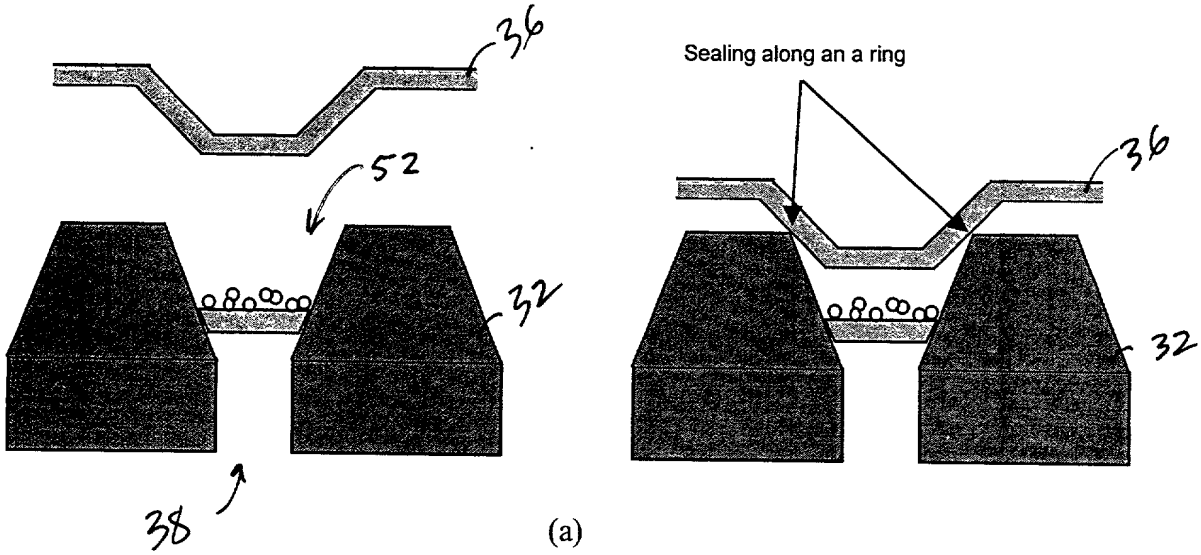
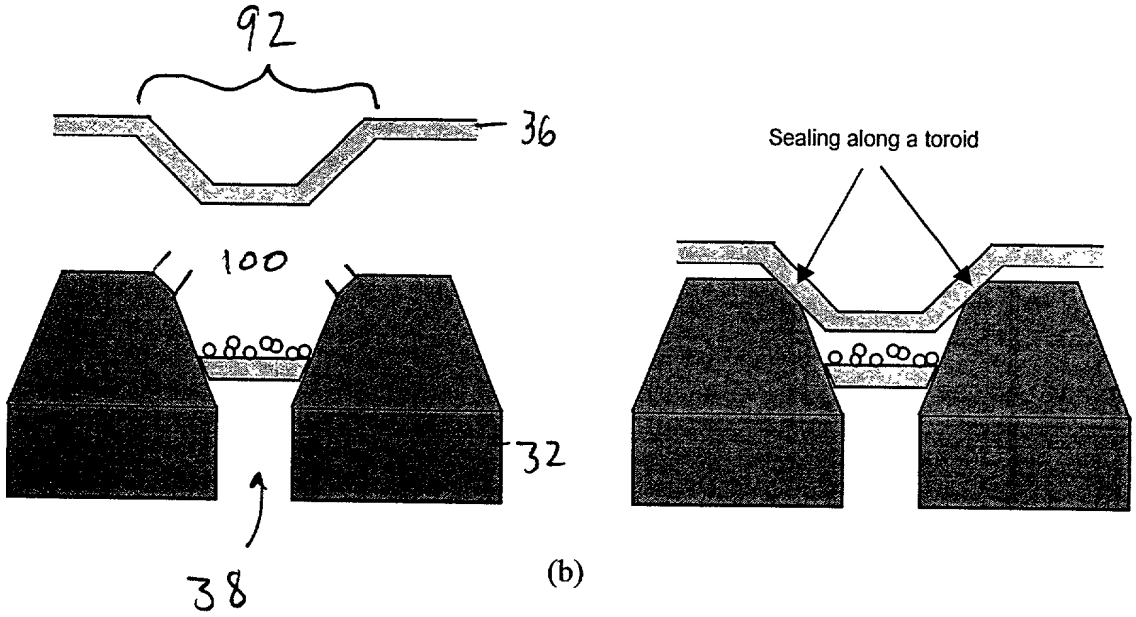


Figure 12


36
52
32
38
Sealing along an a ring
36
32

(a)


92
36
100
32
38
Sealing along a toroid

(b)

Figure 13

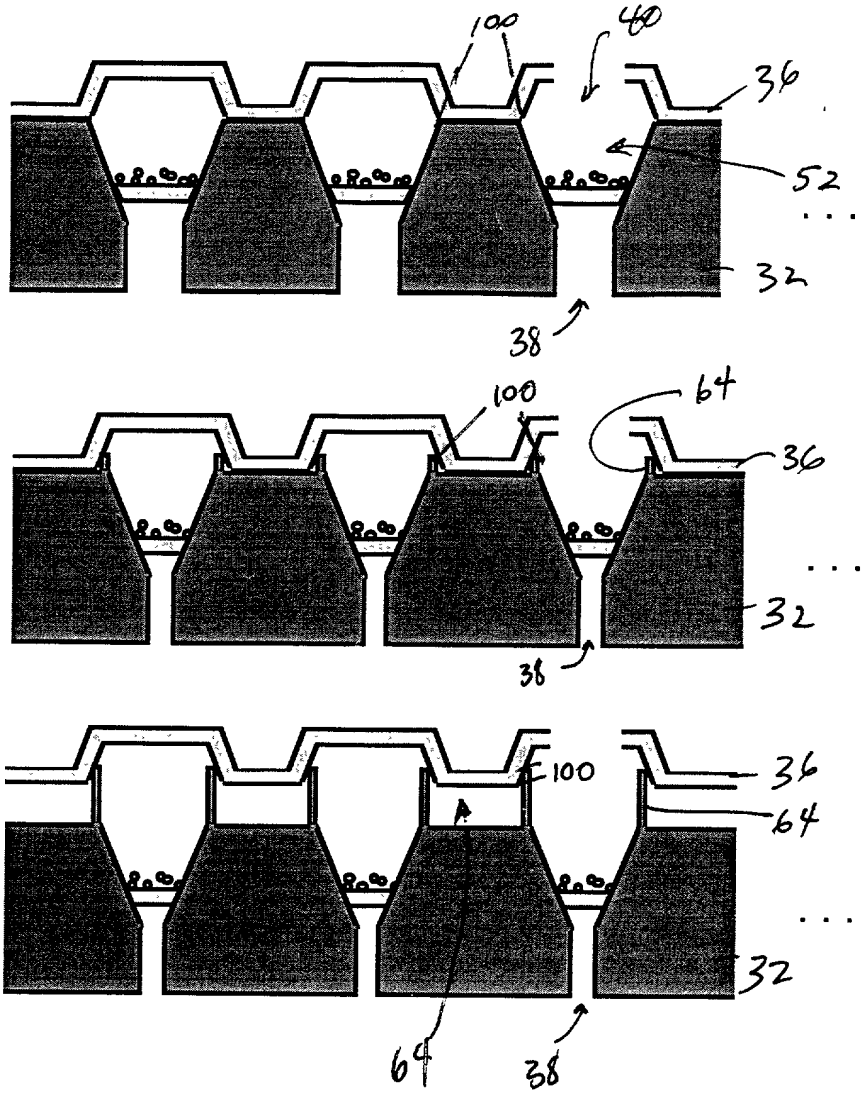
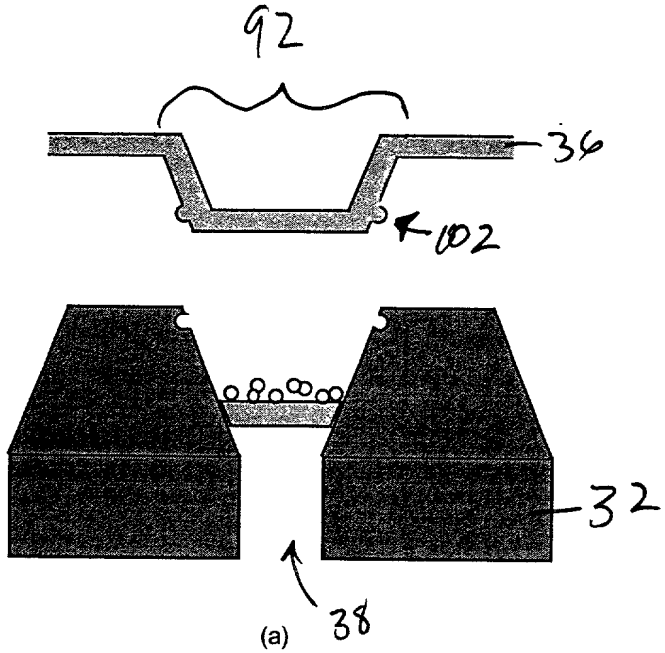
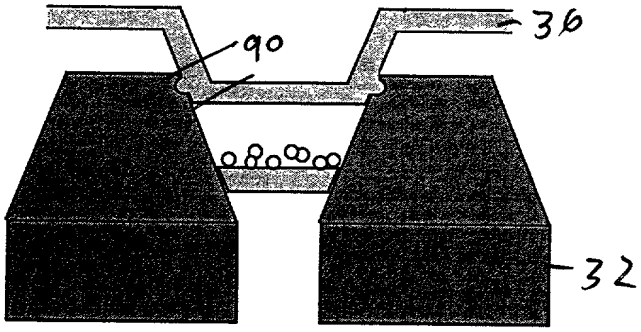


figure 13

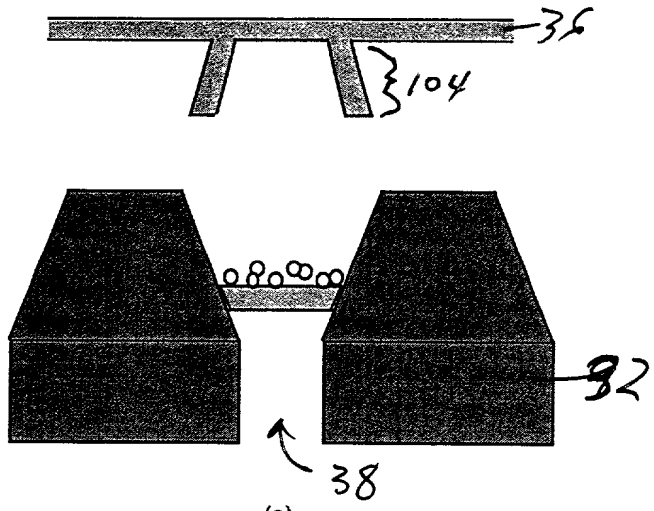


(a)

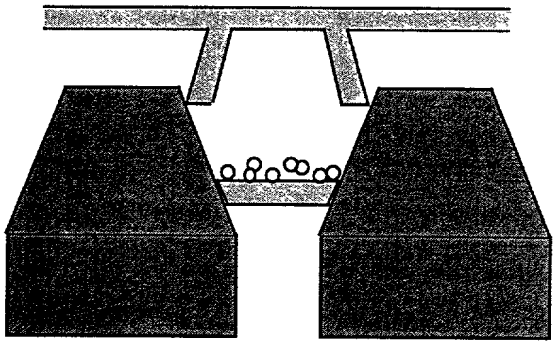


(b)

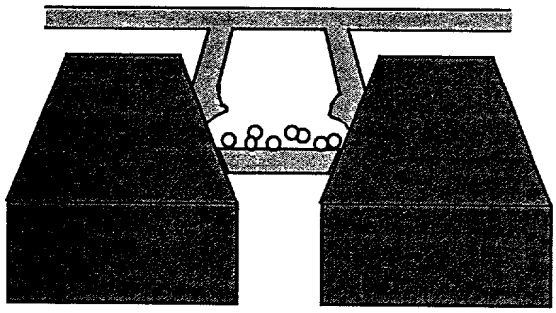
Figure 14



(a)



(b)



(c)

Figure 15

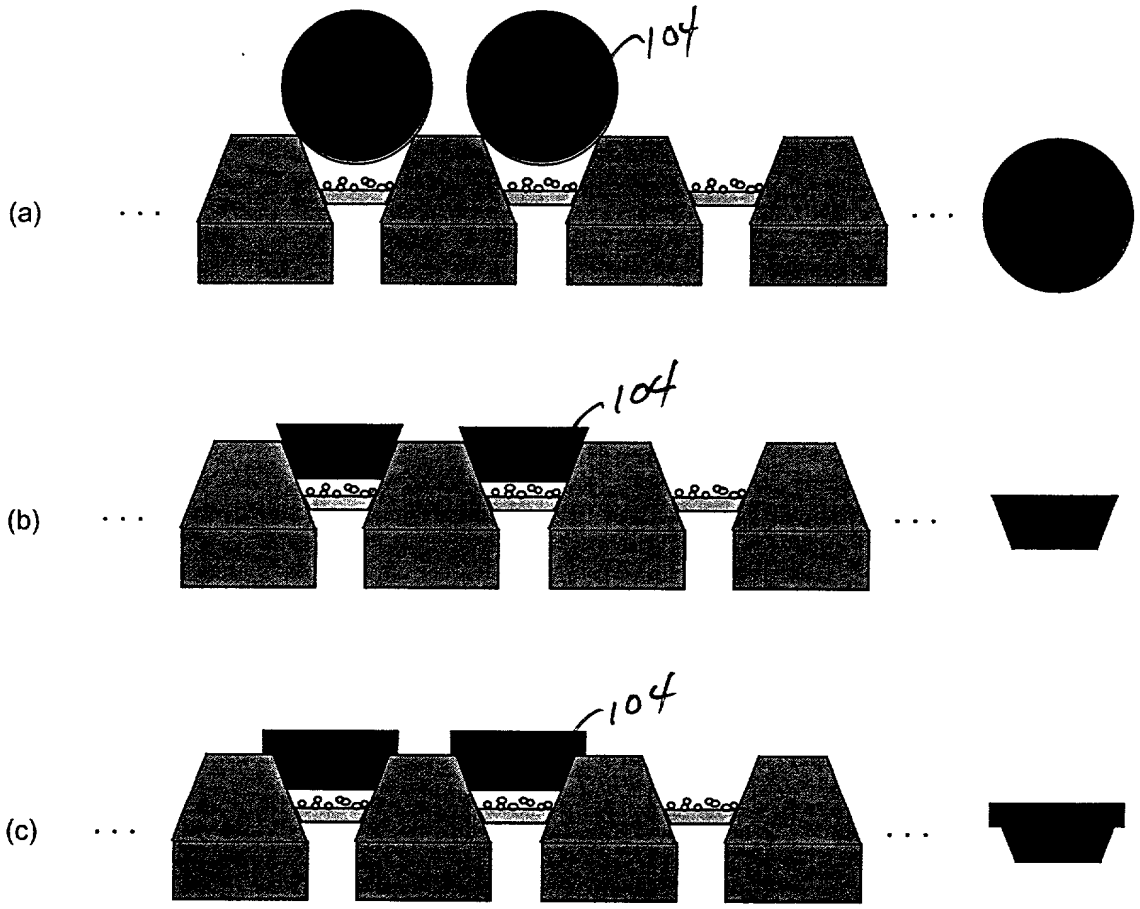


Figure 16

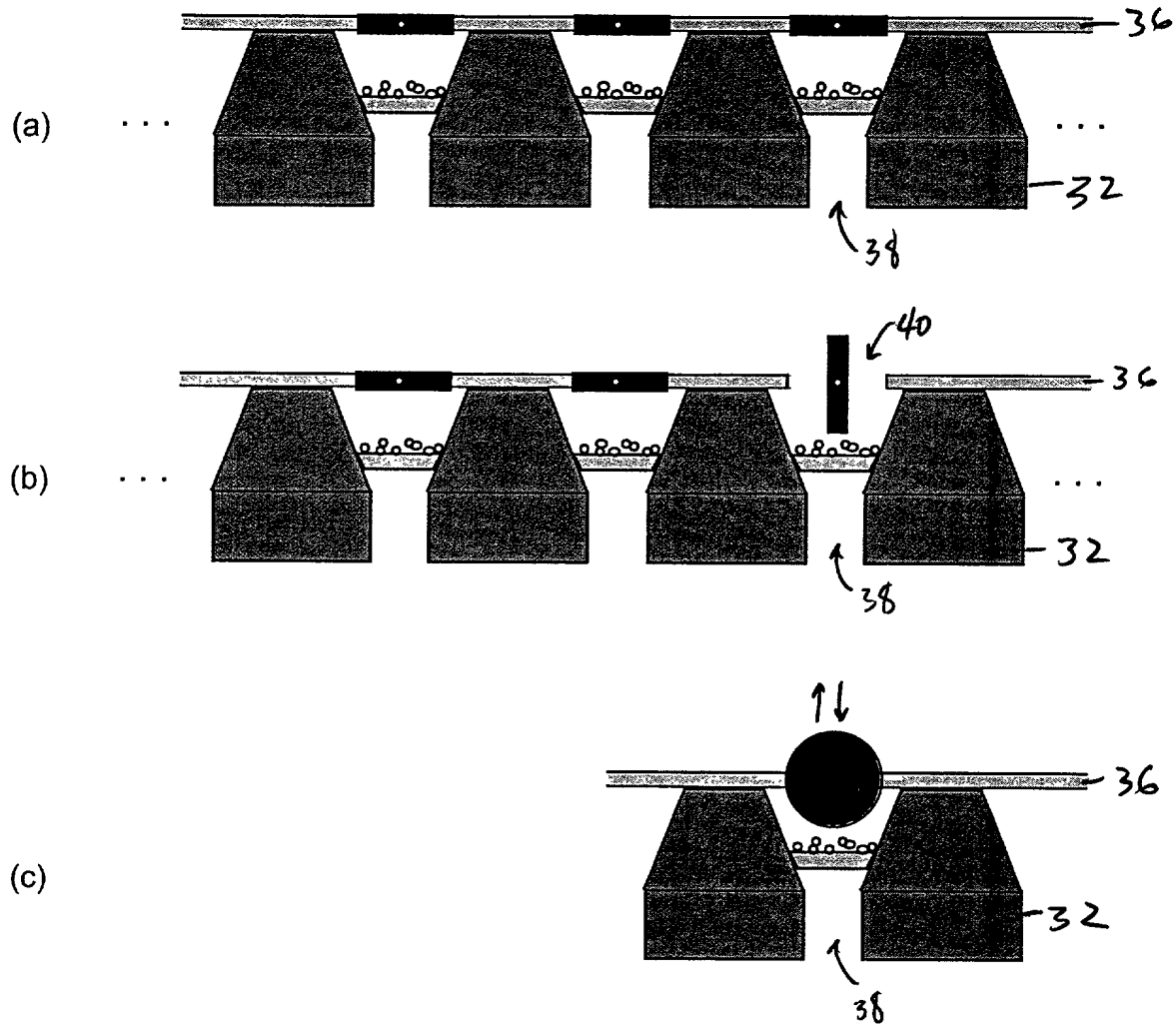


Figure 17

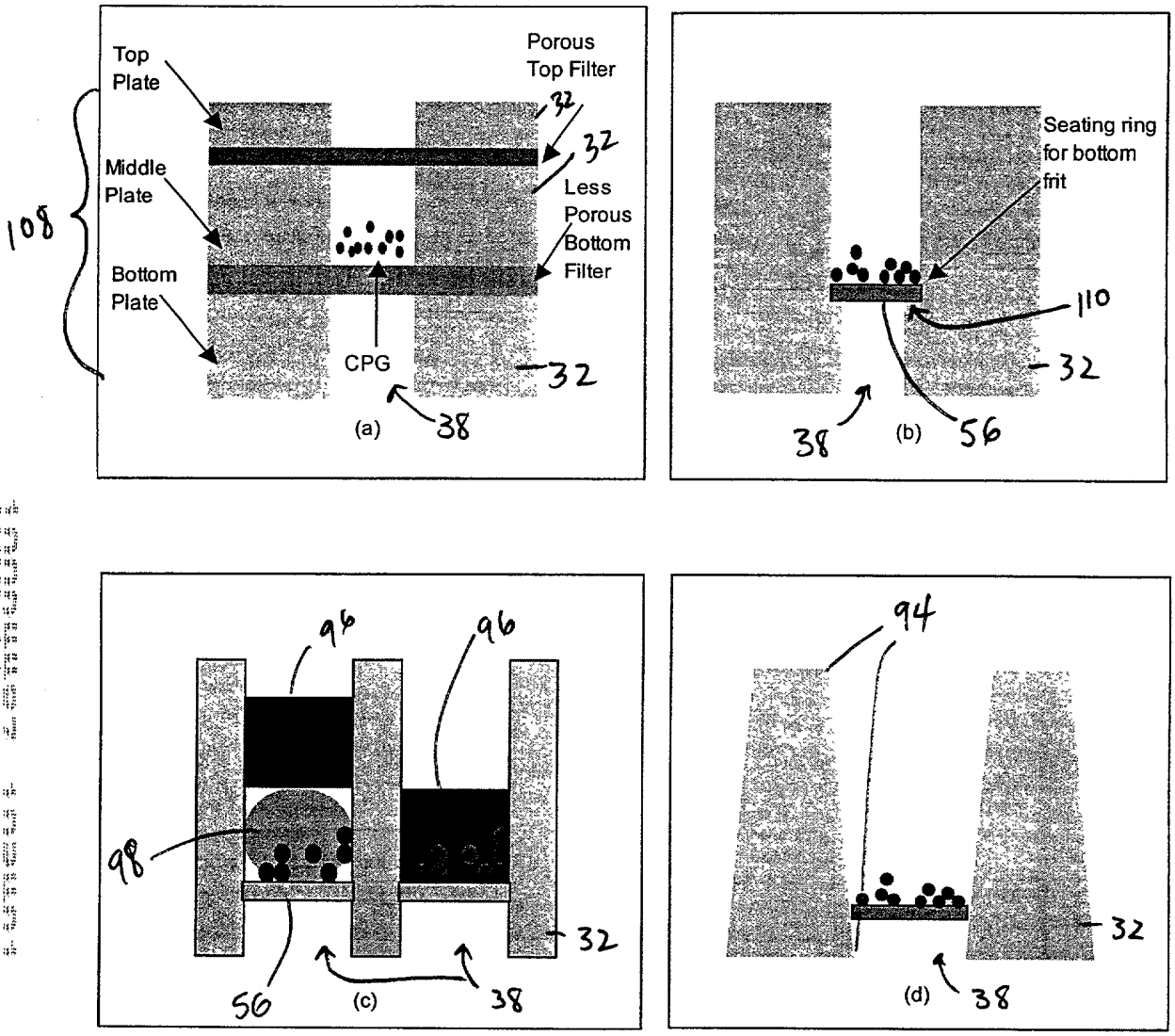

Top Plate
Porous Top Filter
32
32
Middle Plate
Less Porous Bottom Filter
108
Bottom Plate
CPG
32
(a)
38
Seating ring for bottom frit
110
32
38
(b)
56
96
96
98
32
56
(c)
38
94
32
(d)
38

Figure 18

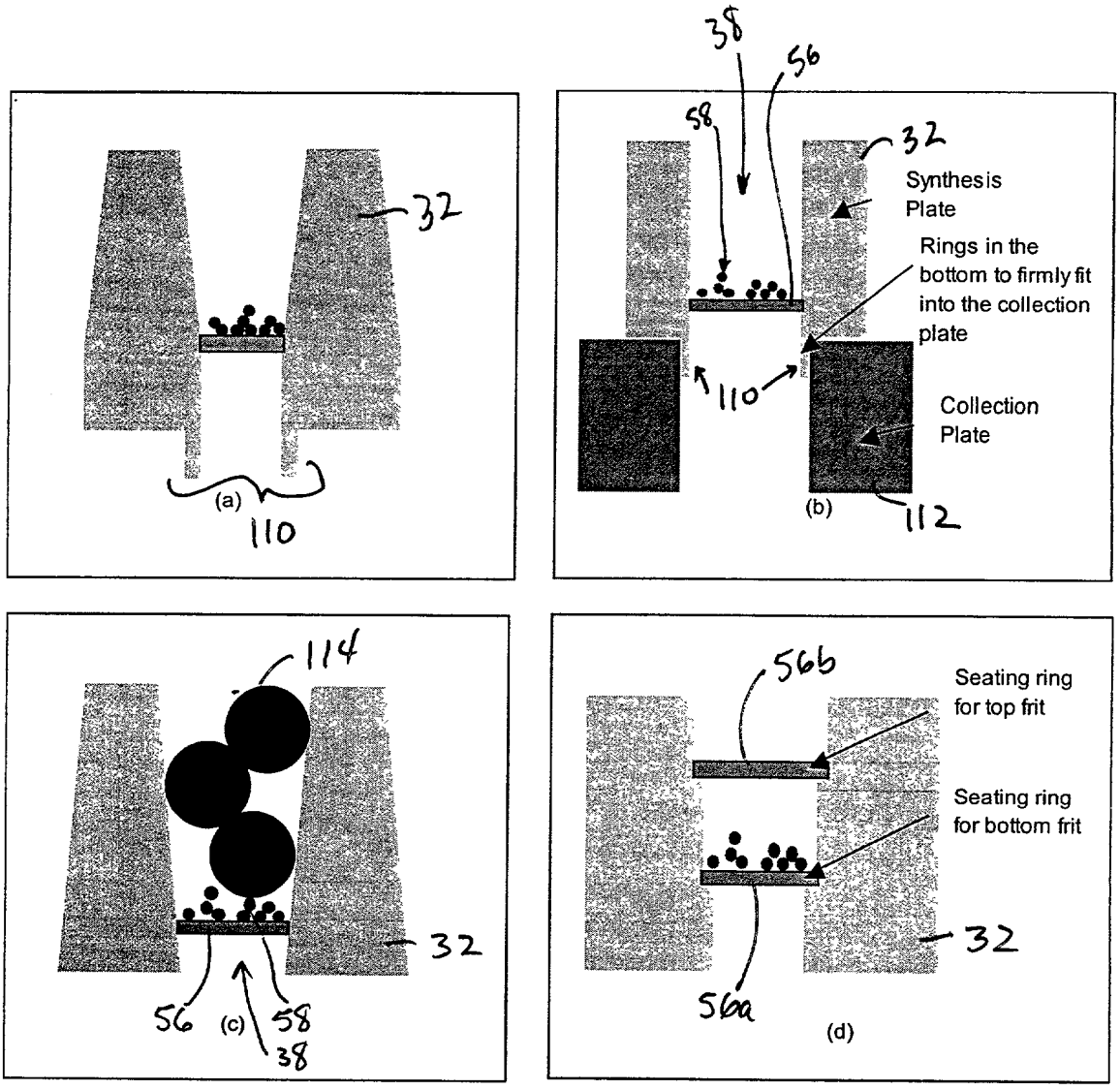

32
(a)
110
38
56
58
32
Synthesis Plate
Rings in the bottom to firmly fit into the collection plate
110
Collection Plate
(b)
112
114
32
56
(c)
58
38
56b
Seating ring for top frit
Seating ring for bottom frit
32
56a
(d)

Figure 19

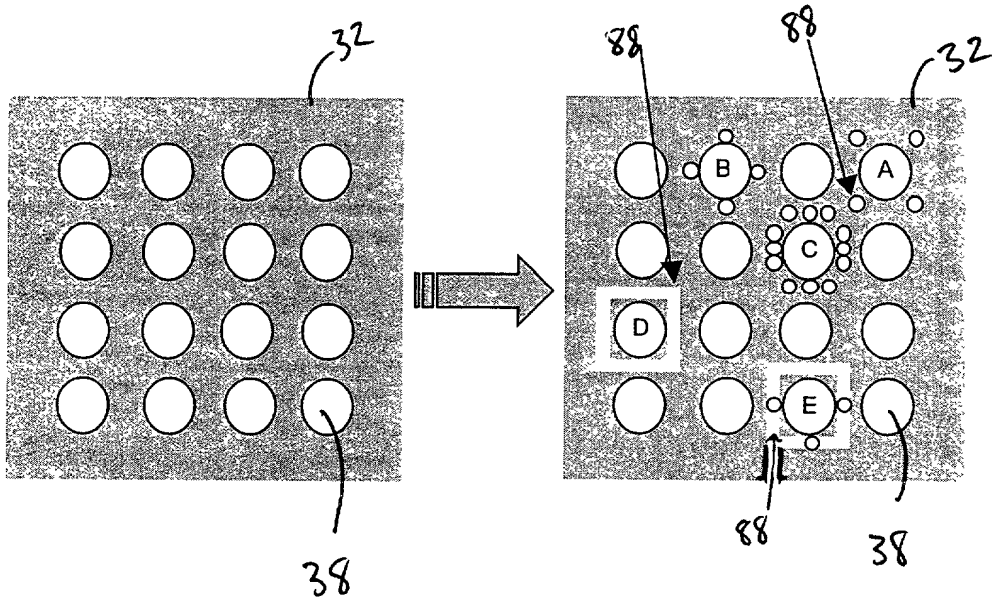
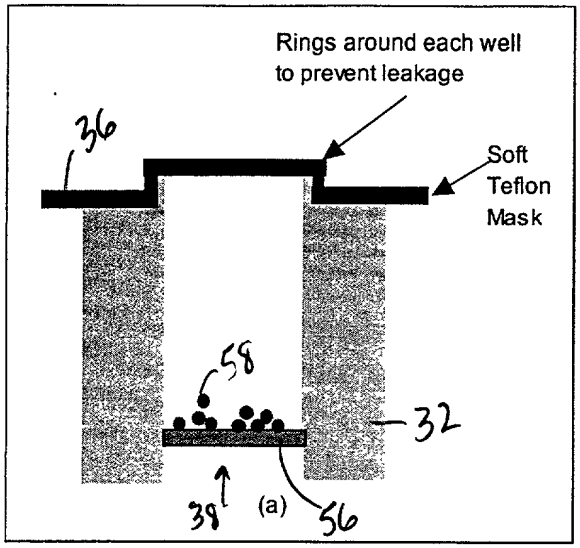
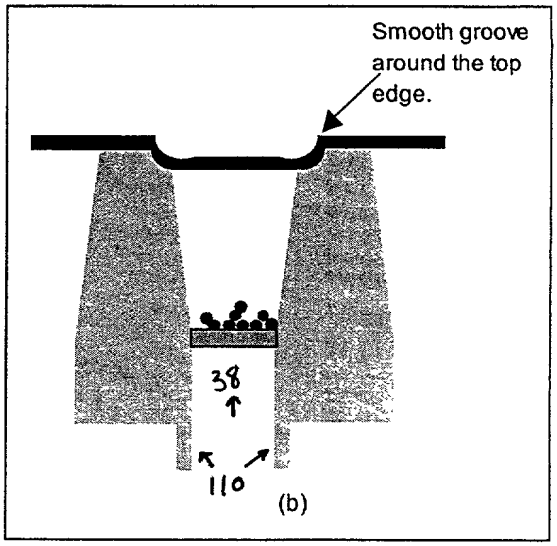


Figure 20


Rings around each well
to prevent leakage
36
Soft
Teflon
Mask
58
32
38
(a)
56


Smooth groove
around the top
edge.
38
110
(b)

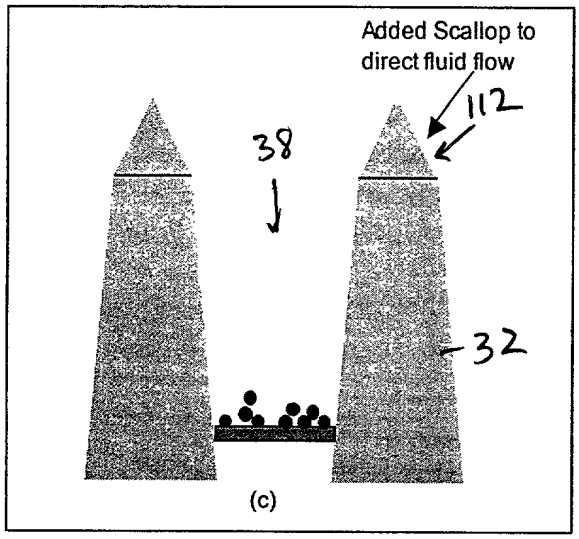

Added Scallop to
direct fluid flow
112
38
32
(c)

Figure 21

Figure 22

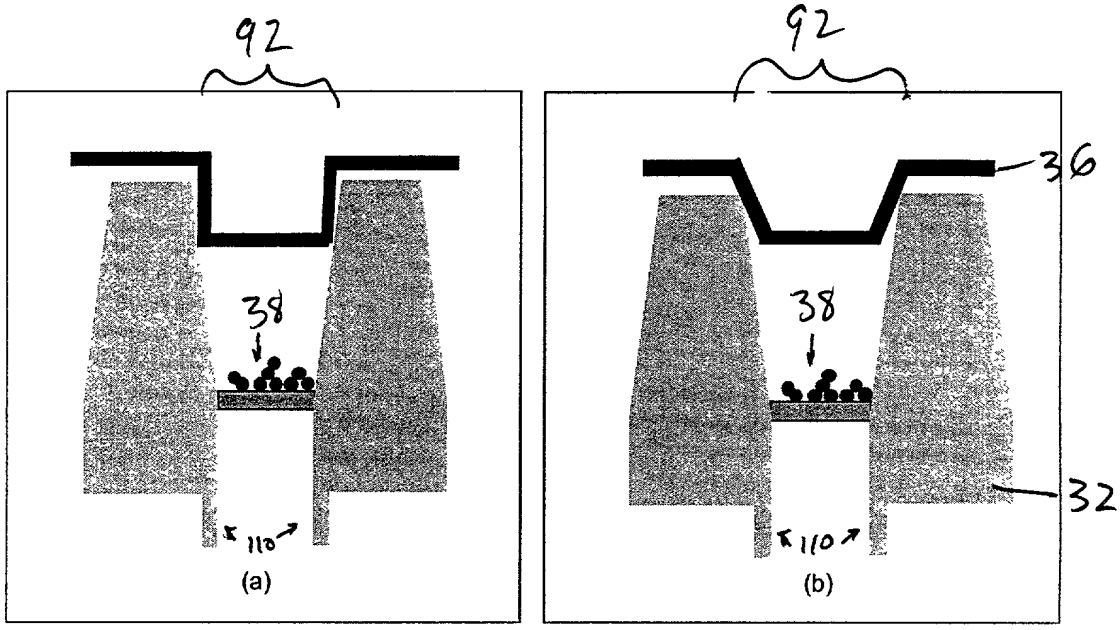


Figure 23

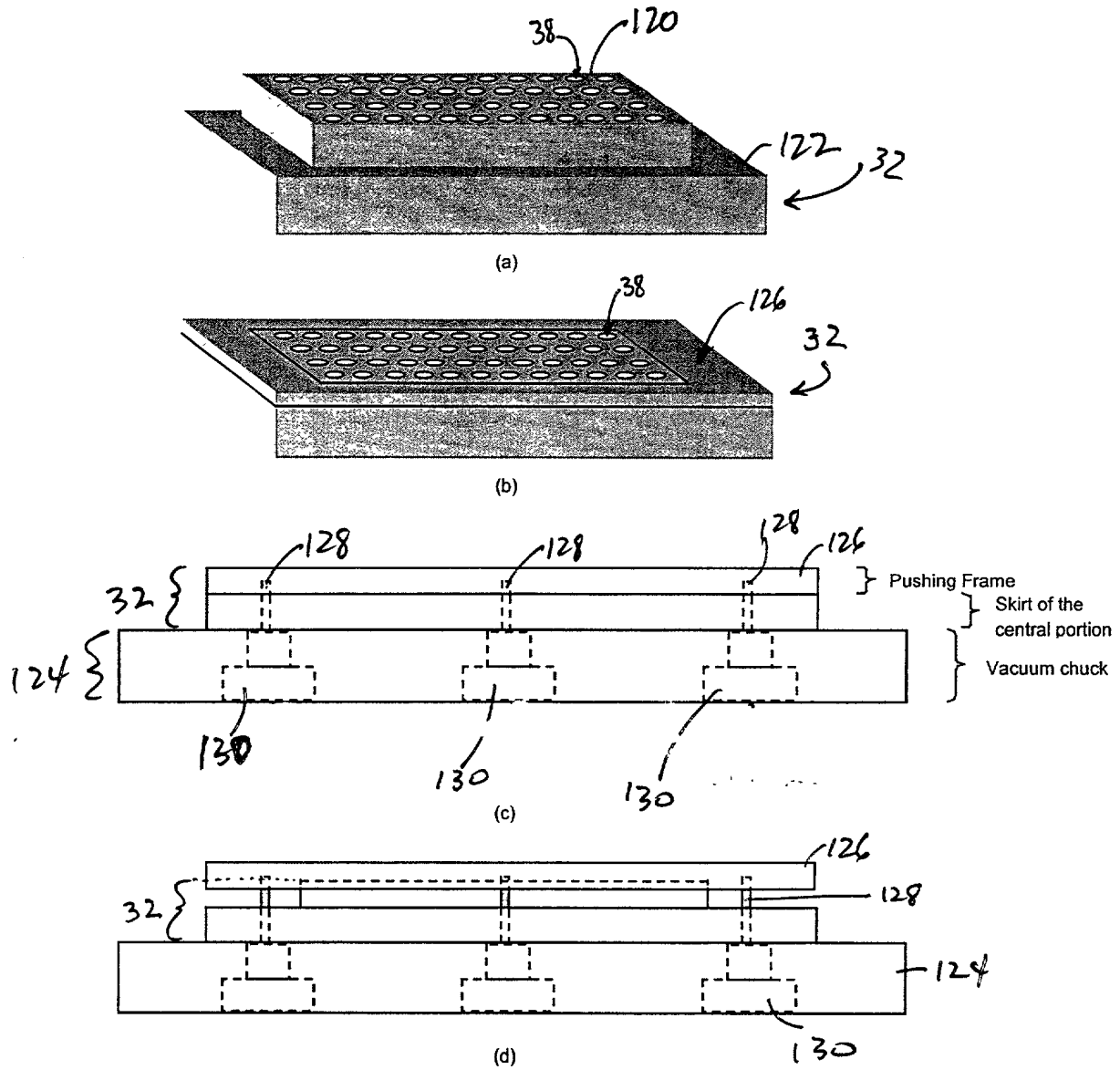

38
120
122
32
(a)
38
126
32
(b)
128
128
128
126
Pushing Frame
Skirt of the central portion
Vacuum chuck
32
124
130
130
130
(c)
126
128
32
124
130
(d)

Figure 24

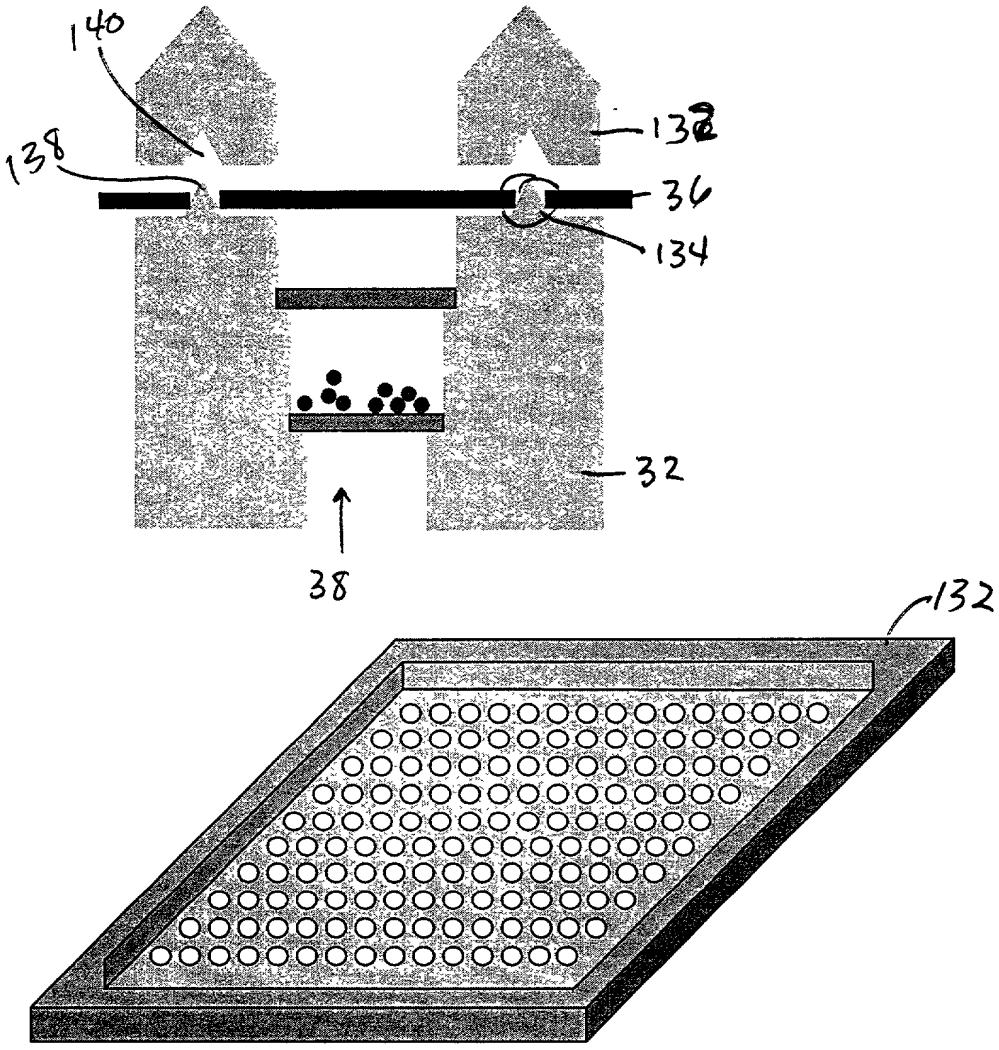


Figure 25

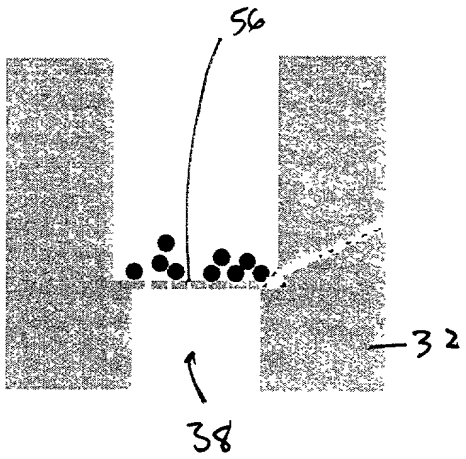


Figure 26

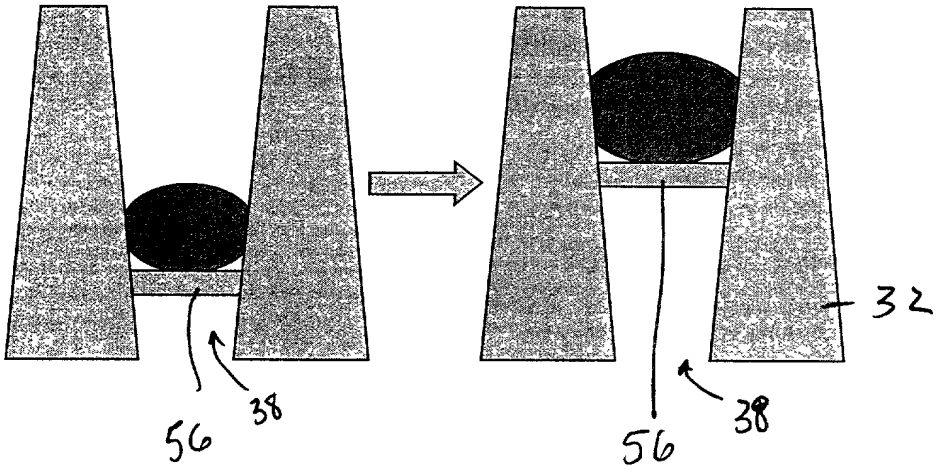


Figure 27

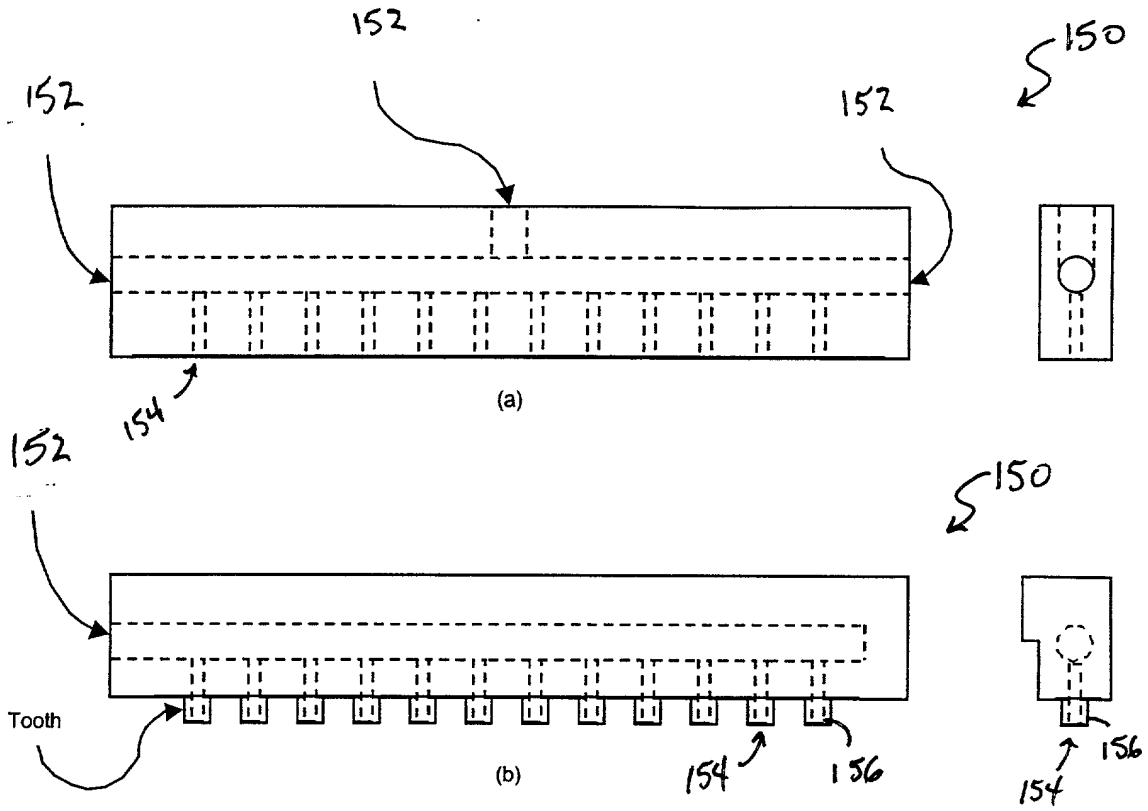
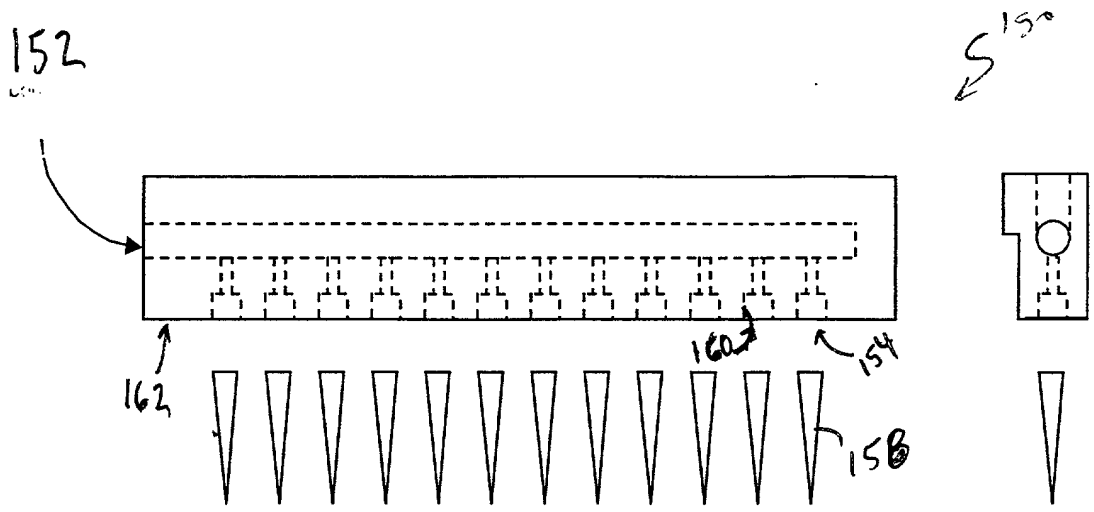
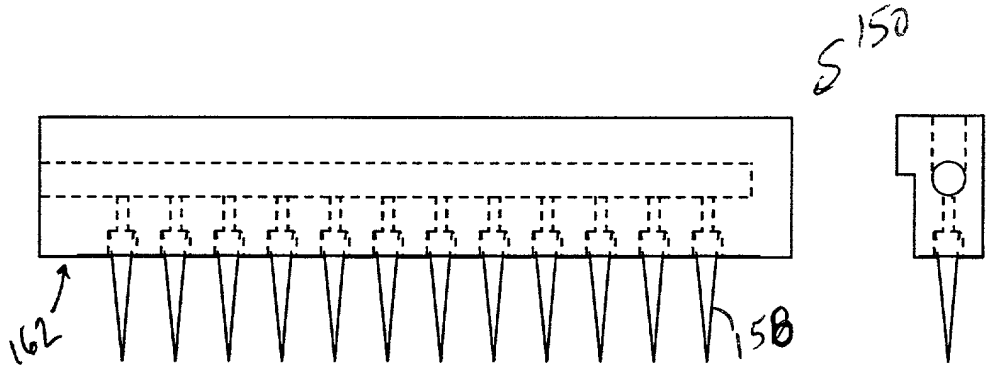


Figure 28



(a)



(b)

Figure 29

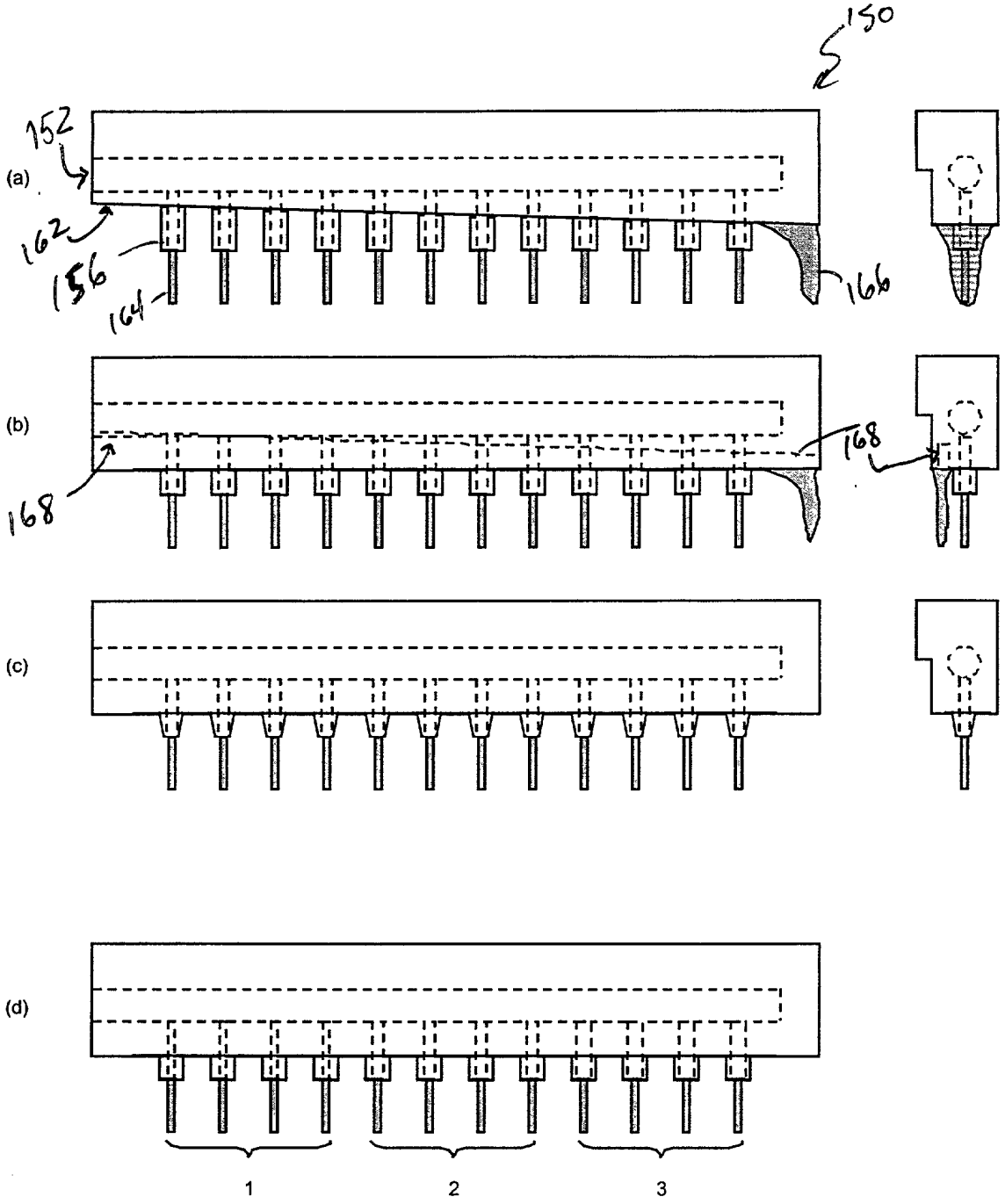


Figure 30

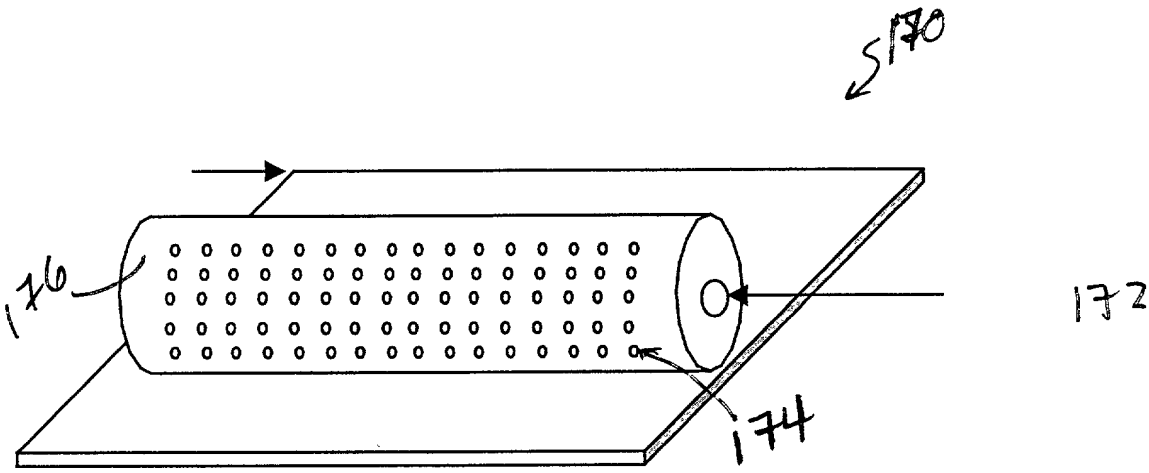


Figure 31

Step		Wait Time (ms)	Prime?	Flush?	Vacuum
0	DEBLOCK	35000	Yes	No	NOT_AT_ALL
1	DEBLOCK	35000	No	No	NOT_AT_ALL
2	DEBLOCK	30000	No	No	FOLLOWING
3	DEBLOCK	30000	No	No	NOT_AT_ALL
4	DEBLOCK	30000	No	No	NOT_AT_ALL
5	DEBLOCK	30000	No	Yes	FOLLOWING
6	ACETONITRILE_WASH	5100	No	No	FOLLOWING
7	ACETONITRILE_WASH	20100	No	No	FOLLOWING
8	ACETONITRILE_WASH	5100	No	No	FOLLOWING
9	ACETONITRILE_WASH	5100	No	No	FOLLOWING
10	COUPLE	35000	Yes	No	NOT_AT_ALL
11	COUPLE	35000	No	NO	FOLLOWING
12	COUPLE	35000	No	Yes	FOLLOWING
13	ACETONITRILE_WASH	5100	No	No	FOLLOWING
14	ACETONITRILE_WASH	20100	No	No	FOLLOWING
15	ACETONITRILE_WASH	5100	No	No	FOLLOWING
16	ACETONITRILE_WASH	5100	No	No	FOLLOWING
17	CAP	30000	Yes	No	NOT_AT_ALL
18	CAP	30000	No	Yes	FOLLOWING
19	ACETONITRILE_WASH	5100	No	No	FOLLOWING
20	ACETONITRILE_WASH	20100	No	No	FOLLOWING
21	ACETONITRILE_WASH	5100	No	No	FOLLOWING
22	ACETONITRILE_WASH	5100	No	No	FOLLOWING
23	OXIDIZE	30000	Yes	No	NOT_AT_ALL
24	OXIDIZE	30000	No	Yes	FOLLOWING
25	ACETONITRILE_WASH	5100	No	No	FOLLOWING
26	ACETONITRILE_WASH	5100	No	No	FOLLOWING
27	ACETONITRILE_WASH	20100	No	No	FOLLOWING
28	ACETONITRILE_WASH	5100	No	No	FOLLOWING
29	ACETONITRILE_WASH	5100	No	No	FOLLOWING
30	ACETONITRILE_WASH	5100	No	No	DURING
31	ACETONITRILE_WASH	100	No	No	DURING
32	ACETONITRILE_WASH	100	No	No	DURING

Figure 32

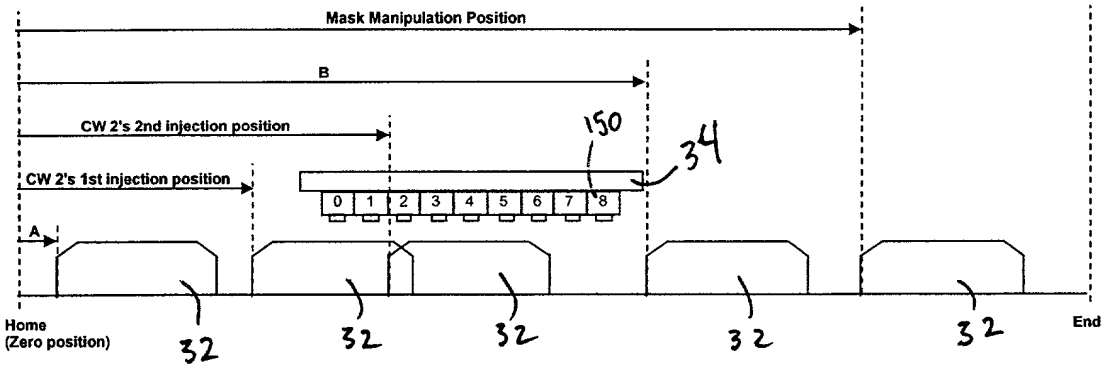

Mask Manipulation Position
B
CW 2's 2nd injection position
CW 2's 1st injection position
A
150
34
0
1
2
3
4
5
6
7
8
Home
(Zero position)
32
32
32
32
32
End

Figure 33

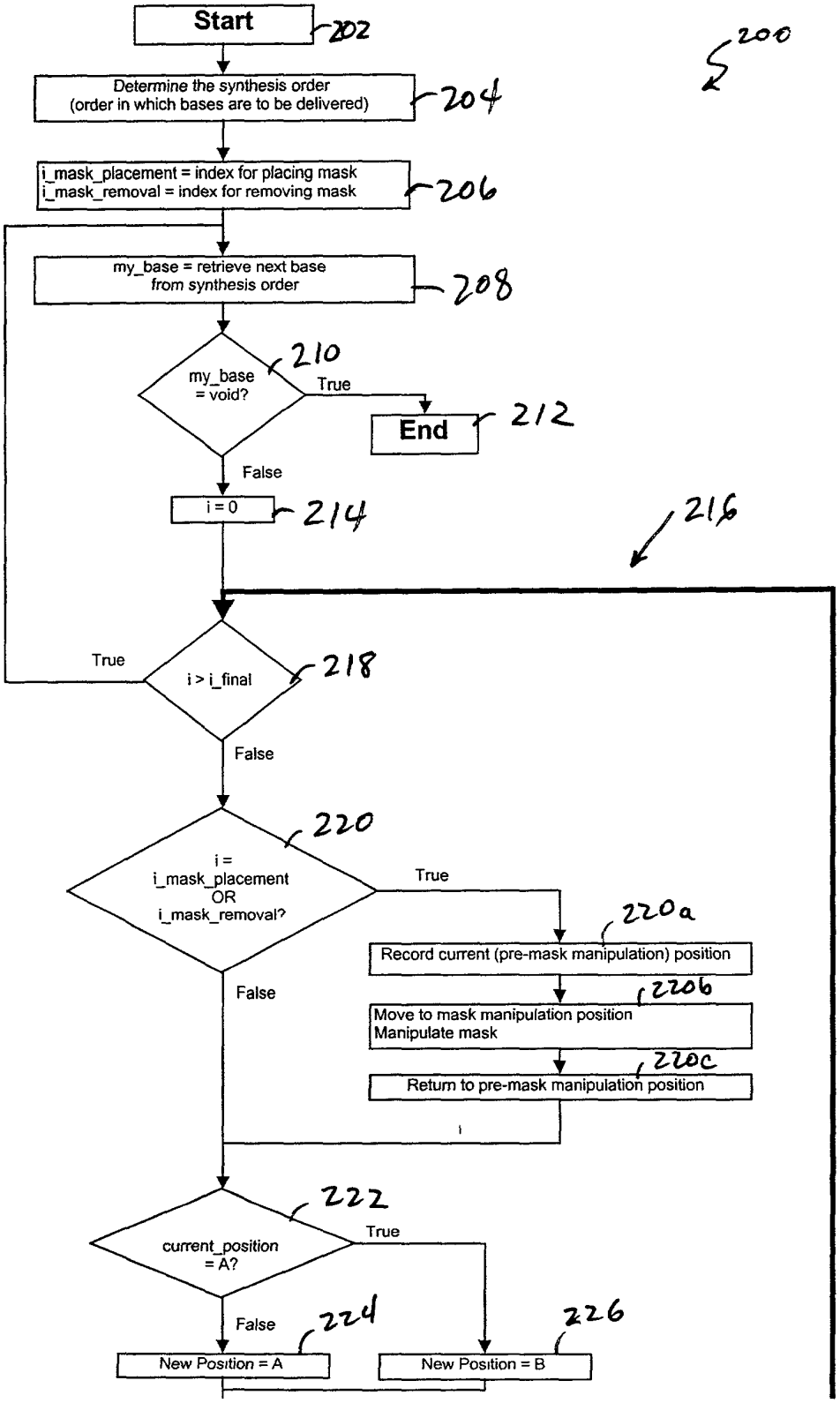

Start
202
200
Determine the synthesis order
(order in which bases are to be delivered)
204
i_mask_placement = index for placing mask
i_mask_removal = index for removing mask
206
my_base = retrieve next base
from synthesis order
208
my_base
= void?
210
True
End
212
False
i = 0
214
216
True
i > i_final
218
False
i =
i_mask_placement
OR
i_mask_removal?
220
True
220a
Record current (pre-mask manipulation) position
220b
Move to mask manipulation position
Manipulate mask
220c
Return to pre-mask manipulation position
False
222
current_position
= A?
True
False
224
New Position = A
226
New Position = B

Figure 34

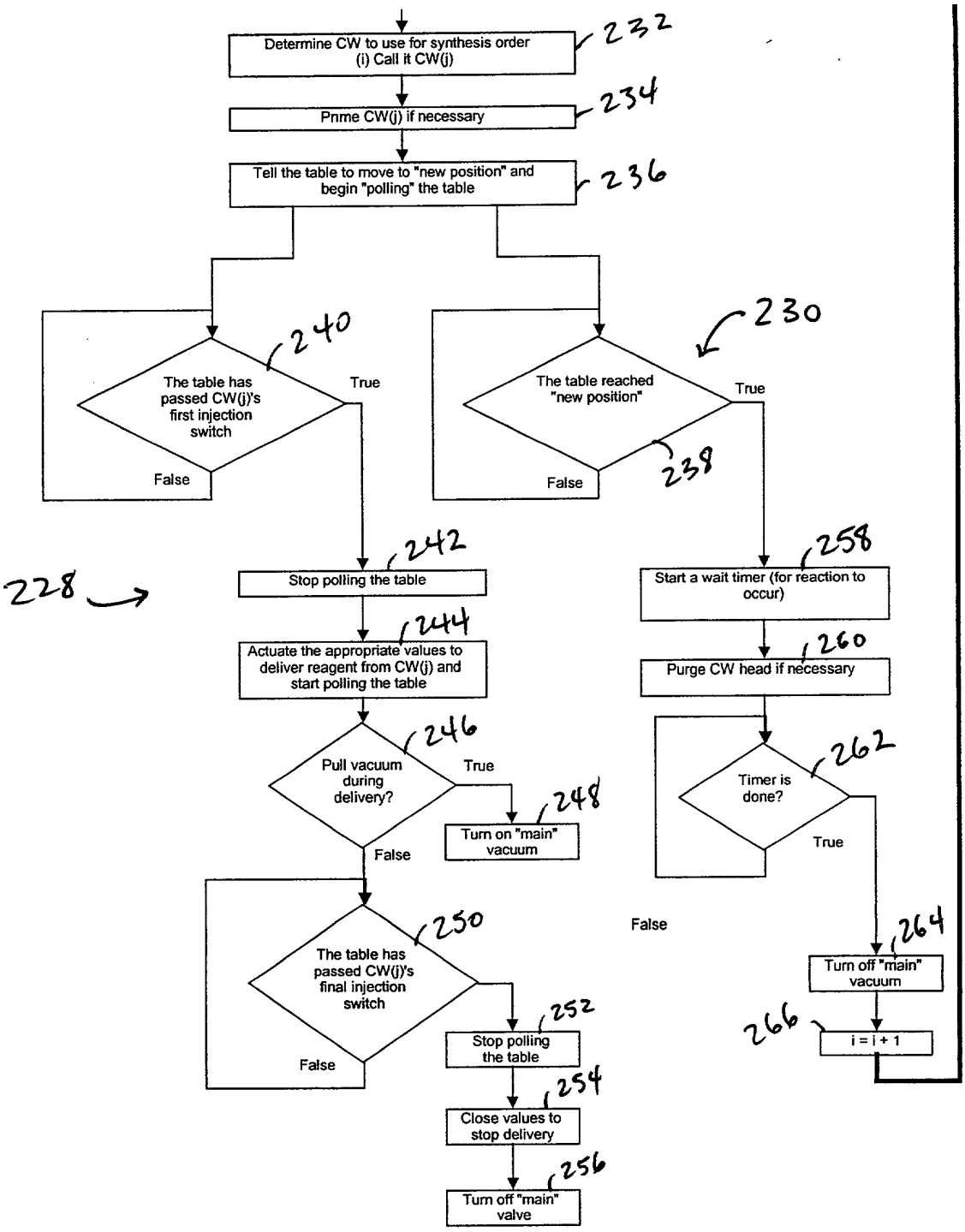

Determine CW to use for synthesis order (i) Call it CW(j)
232
Prime CW(j) if necessary
234
Tell the table to move to "new position" and begin "polling" the table
236
230
The table has passed CW(j)'s first injection switch
240
True
False
The table reached "new position"
True
238
False
228
Stop polling the table
242
Actuate the appropriate values to deliver reagent from CW(j) and start polling the table
244
Pull vacuum during delivery?
246
True
Turn on "main" vacuum
248
False
The table has passed CW(j)'s final injection switch
250
False
Stop polling the table
252
Close values to stop delivery
254
Turn off "main" valve
256
Start a wait timer (for reaction to occur)
258
Purge CW head if necessary
260
Timer is done?
262
True
False
Turn off "main" vacuum
264
i = i + 1
266

Figure 35

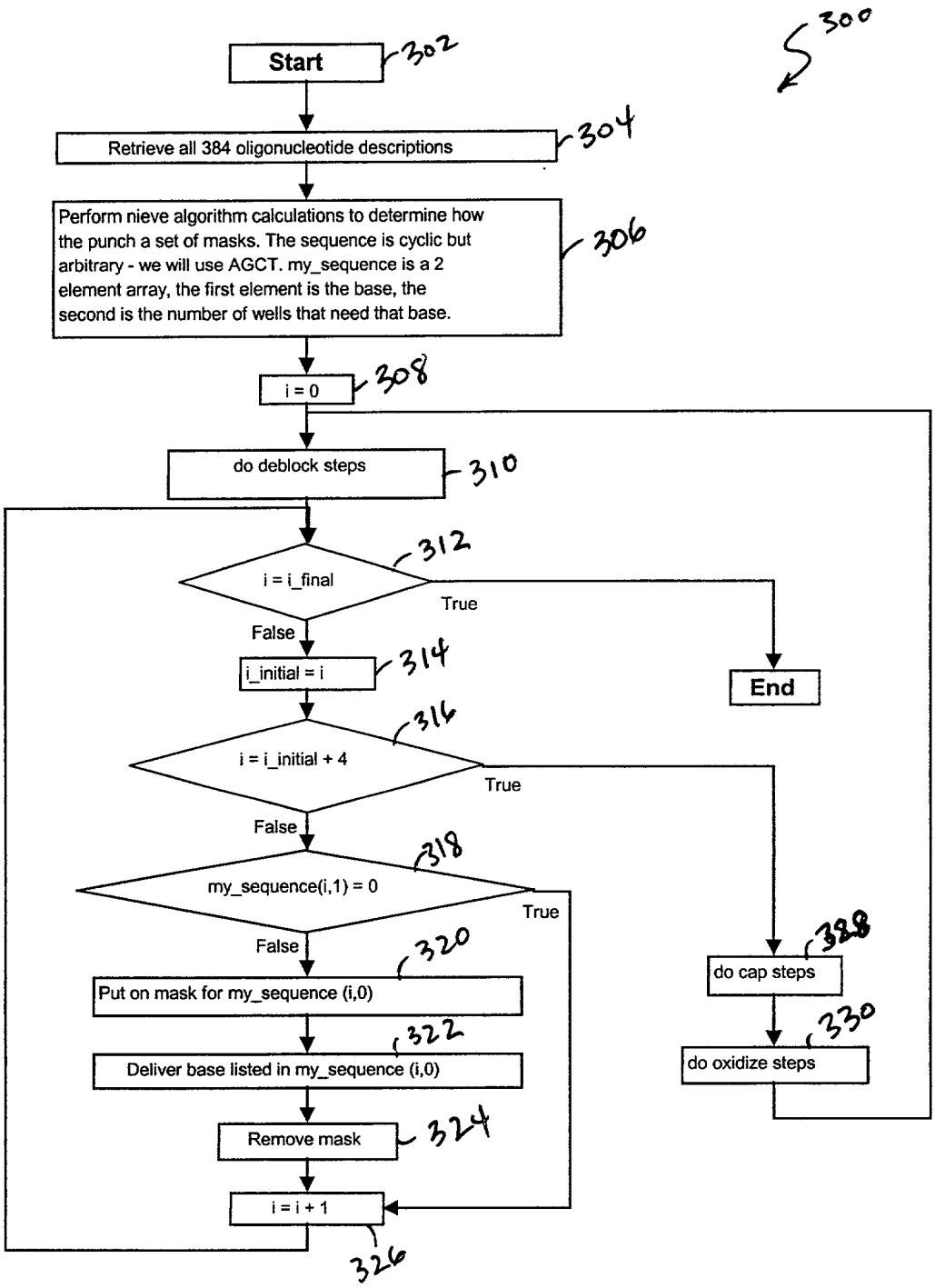
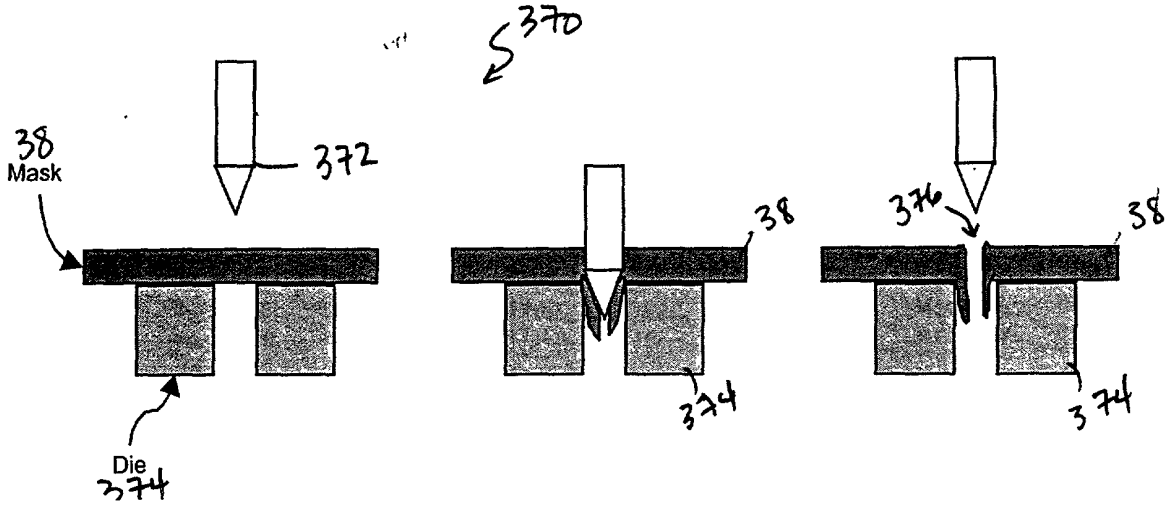

300
Start
302
Retrieve all 384 oligonucleotide descriptions
304
Perform nieve algorithm calculations to determine how the punch a set of masks. The sequence is cyclic but arbitrary - we will use AGCT. my_sequence is a 2 element array, the first element is the base, the second is the number of wells that need that base.
306
i = 0
308
do deblock steps
310
i = i_final
312
True
False
End
i_initial = i
314
i = i_initial + 4
316
True
False
my_sequence(i,1) = 0
318
True
False
do cap steps
328
Put on mask for my_sequence (i,0)
320
do oxidize steps
330
Deliver base listed in my_sequence (i,0)
322
Remove mask
324
i = i + 1
326

Figure 36

Figure 37



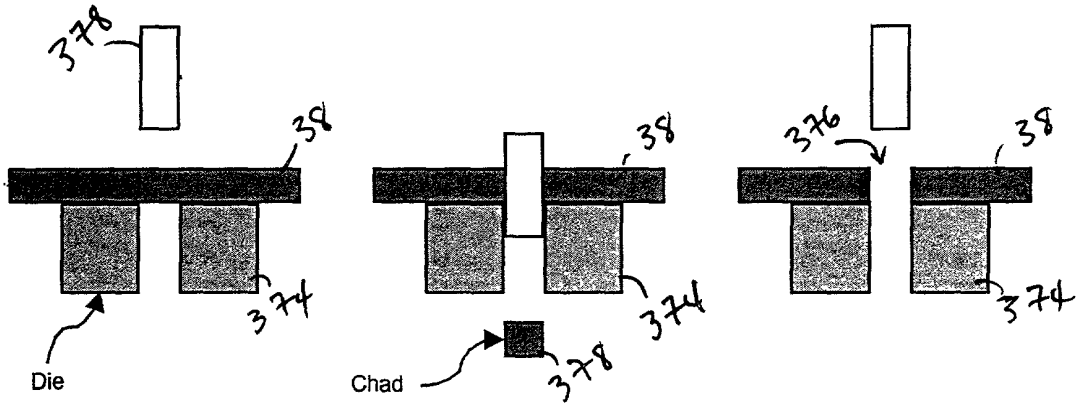
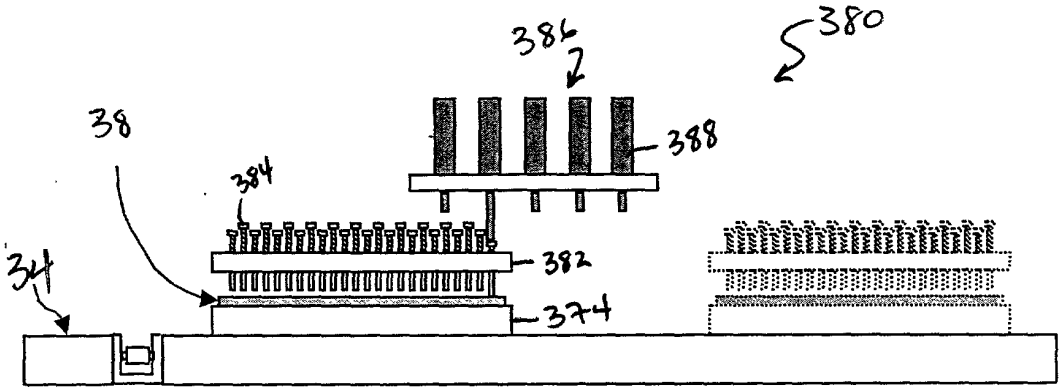
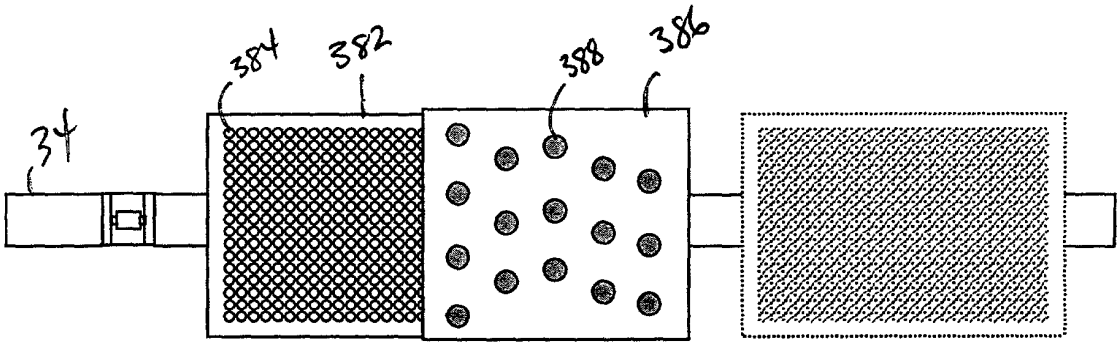


Figure 38



(a)



(b)

Figure 39

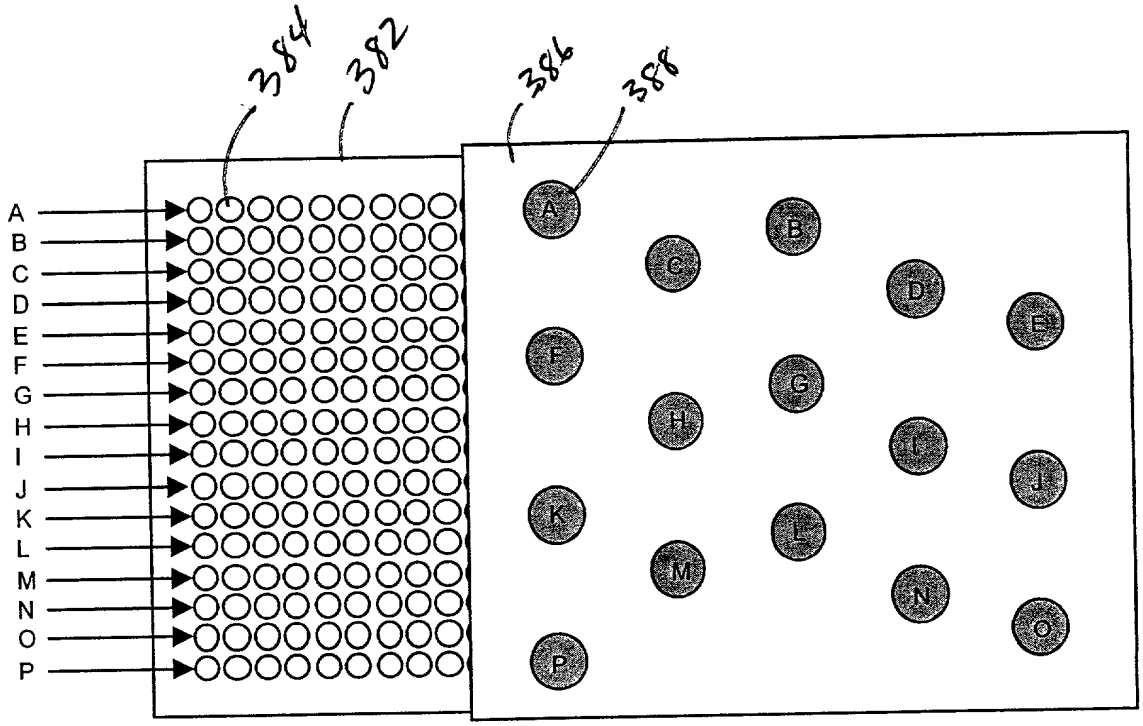


Figure 40

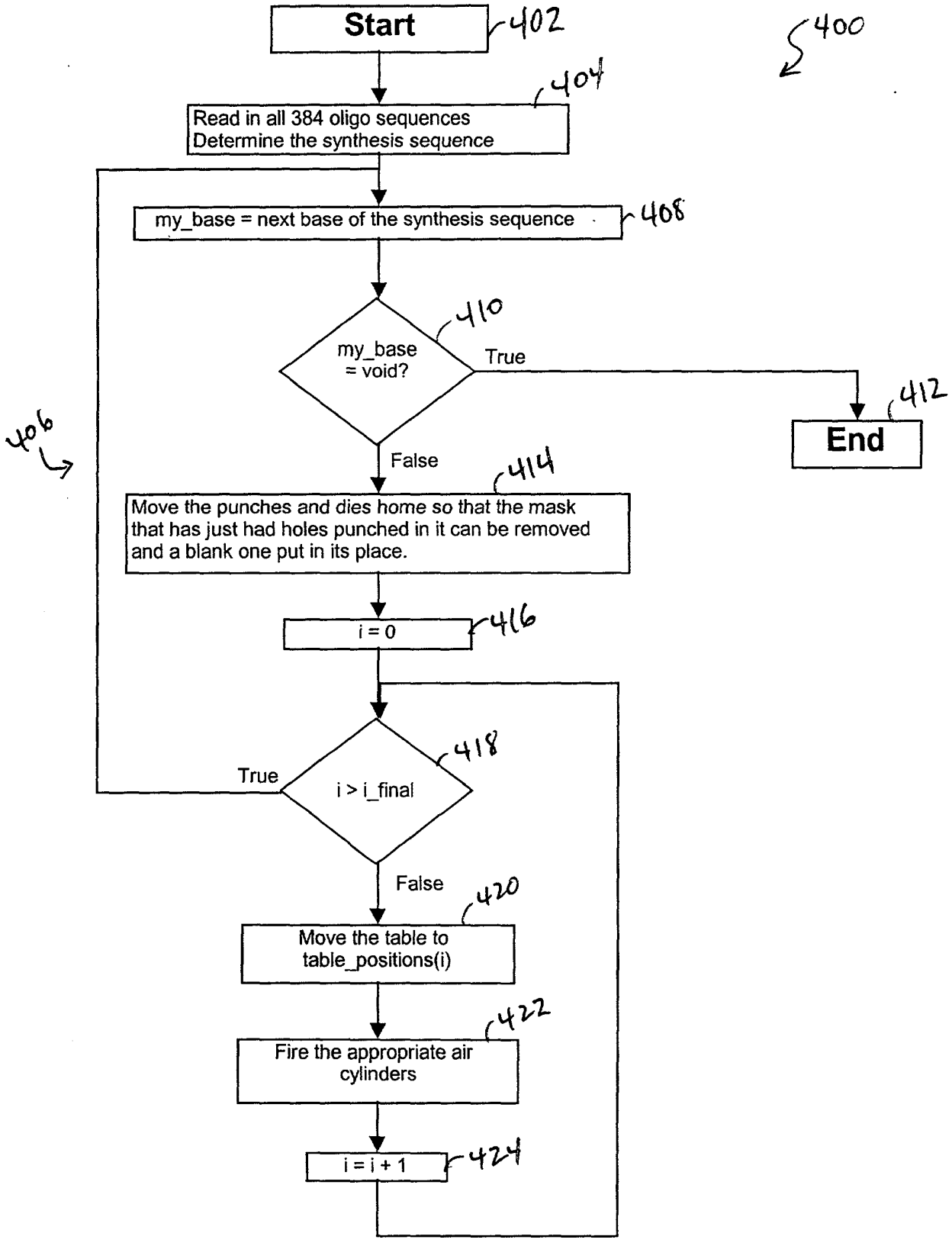

Start
402
400
404
Read in all 384 oligo sequences
Determine the synthesis sequence
my_base = next base of the synthesis sequence
408
410
my_base
= void?
True
412
End
406
False
414
Move the punches and dies home so that the mask
that has just had holes punched in it can be removed
and a blank one put in its place.
i = 0
416
418
True
i > i_final
False
420
Move the table to
table_positions(i)
422
Fire the appropriate air
cylinders
i = i + 1
424

Figure 41

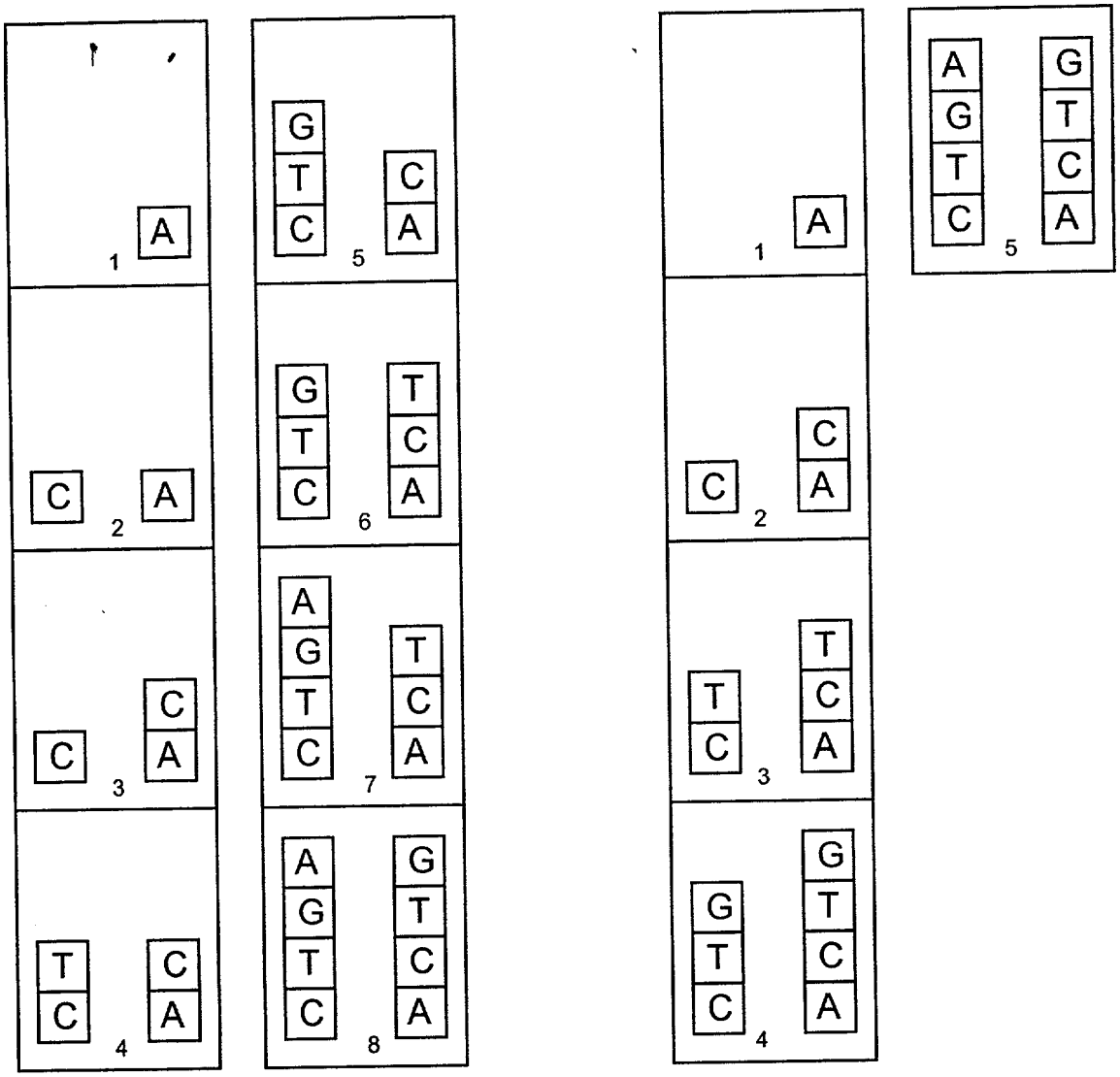

A
1
C
A
2
C
C
A
3
T
C
C
A
4
G
T
C
C
A
5
G
T
C
T
C
A
6
A
G
T
C
T
C
A
7
A
G
T
C
G
T
C
A
8
(a)
A
1
C
A
2
T
C
T
C
A
3
G
T
C
G
T
C
A
4
(b)
A
G
T
C
G
T
C
A
5

Figure 42

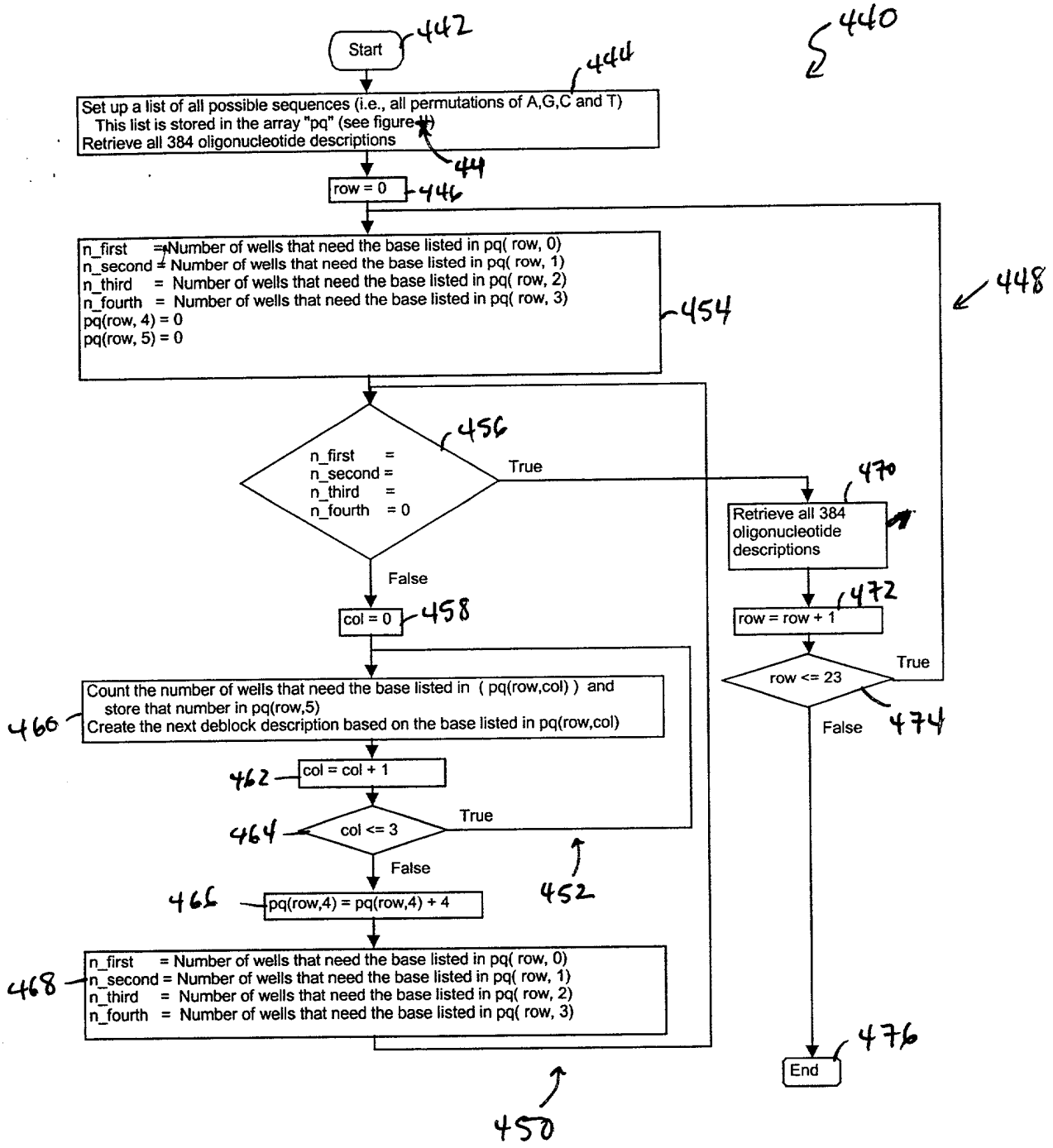
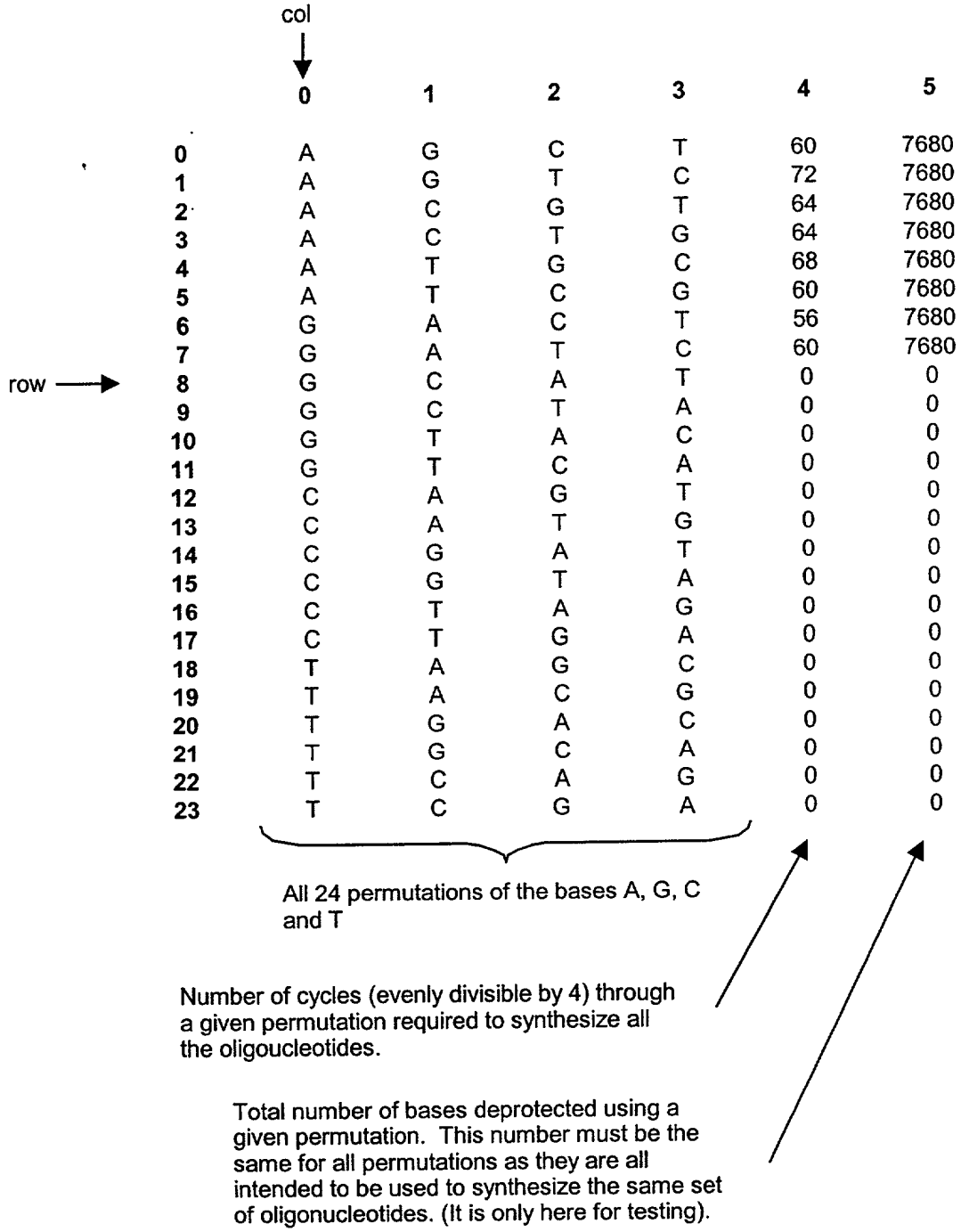

Start
442
440
444
Set up a list of all possible sequences (i.e., all permutations of A,G,C and T)
This list is stored in the array "pq" (see figure 44)
Retrieve all 384 oligonucleotide descriptions
row = 0
446
n_first = Number of wells that need the base listed in pq(row, 0)
n_second = Number of wells that need the base listed in pq(row, 1)
n_third = Number of wells that need the base listed in pq(row, 2)
n_fourth = Number of wells that need the base listed in pq(row, 3)
pq(row, 4) = 0
pq(row, 5) = 0
454
448
456
n_first =
n_second =
n_third =
n_fourth = 0
True
470
Retrieve all 384 oligonucleotide descriptions
False
col = 0
458
472
row = row + 1
True
row <= 23
False
474
Count the number of wells that need the base listed in (pq(row,col)) and store that number in pq(row,5)
Create the next deblock description based on the base listed in pq(row,col)
466
462
col = col + 1
464
col <= 3
True
False
452
465
pq(row,4) = pq(row,4) + 4
n_first = Number of wells that need the base listed in pq(row, 0)
n_second = Number of wells that need the base listed in pq(row, 1)
n_third = Number of wells that need the base listed in pq(row, 2)
n_fourth = Number of wells that need the base listed in pq(row, 3)
468
476
End
450

Figure 43



	0	1	2	3	4	5
0	A	G	C	T	60	7680
1	A	G	T	C	72	7680
2	A	C	G	T	64	7680
3	A	C	T	G	64	7680
4	A	T	G	C	68	7680
5	A	T	C	G	60	7680
6	G	A	C	T	56	7680
7	G	A	T	C	60	7680
8	G	C	A	T	0	0
9	G	C	T	A	0	0
10	G	T	A	C	0	0
11	G	T	C	A	0	0
12	C	A	G	T	0	0
13	C	A	T	G	0	0
14	C	G	A	T	0	0
15	C	G	T	A	0	0
16	C	T	A	G	0	0
17	C	T	G	A	0	0
18	T	A	G	C	0	0
19	T	A	C	G	0	0
20	T	G	A	C	0	0
21	T	G	C	A	0	0
22	T	C	A	G	0	0
23	T	C	G	A	0	0

Figure 44

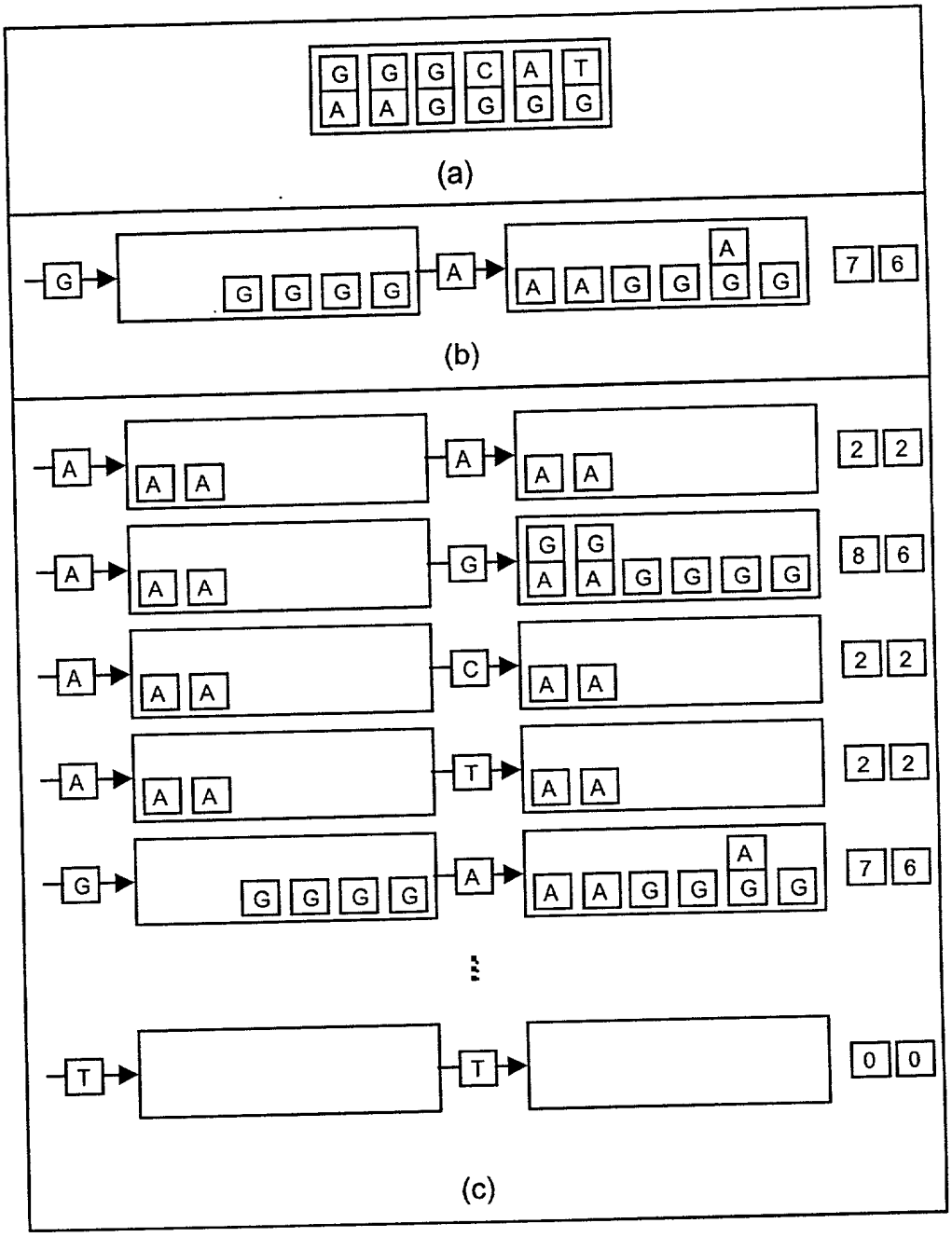

G G G C A T
A A G G G G
(a)
G
G G G G
A
A A G G A G G
7 6
(b)
A
A A
A
A A
2 2
A
A A
G
G G
A A G G G G
8 6
A
A A
C
A A
2 2
A
A A
T
A A
2 2
G
G G G G
A
A A G G A G G
7 6
T
T
0 0
(c)

Figure 45

Sequence		# of coupling reactions			# of oligos coupled		
First Base	Second Base	First Base	Second Base	Total	First pass	Unique on second pass	Total
G	A	4	3	7	4	2	6

(a)

Permutations		# of coupling reactions			# of oligos coupled		
First Base	Second Base	First Base	Second Base	Total	First pass	Unique on second pass	Total
A	A	2	0	2	2	0	2
A	G	2	6	8	2	4	6
A	C	2	0	2	2	0	2
A	T	2	0	2	2	0	2
G	A	4	3	7	4	2	6
G	G	4	1	5	4	0	4
G	C	4	1	5	4	0	4
G	T	4	1	5	4	0	4
C	A	0	2	2	0	2	2
C	G	0	4	4	0	4	4
C	C	0	0	0	0	0	0
C	T	0	0	0	0	0	0
T	A	0	2	2	0	2	2
T	G	0	4	4	0	4	4
T	C	0	0	0	0	0	0
T	T	0	0	0	0	0	0

(b)

Figure 46

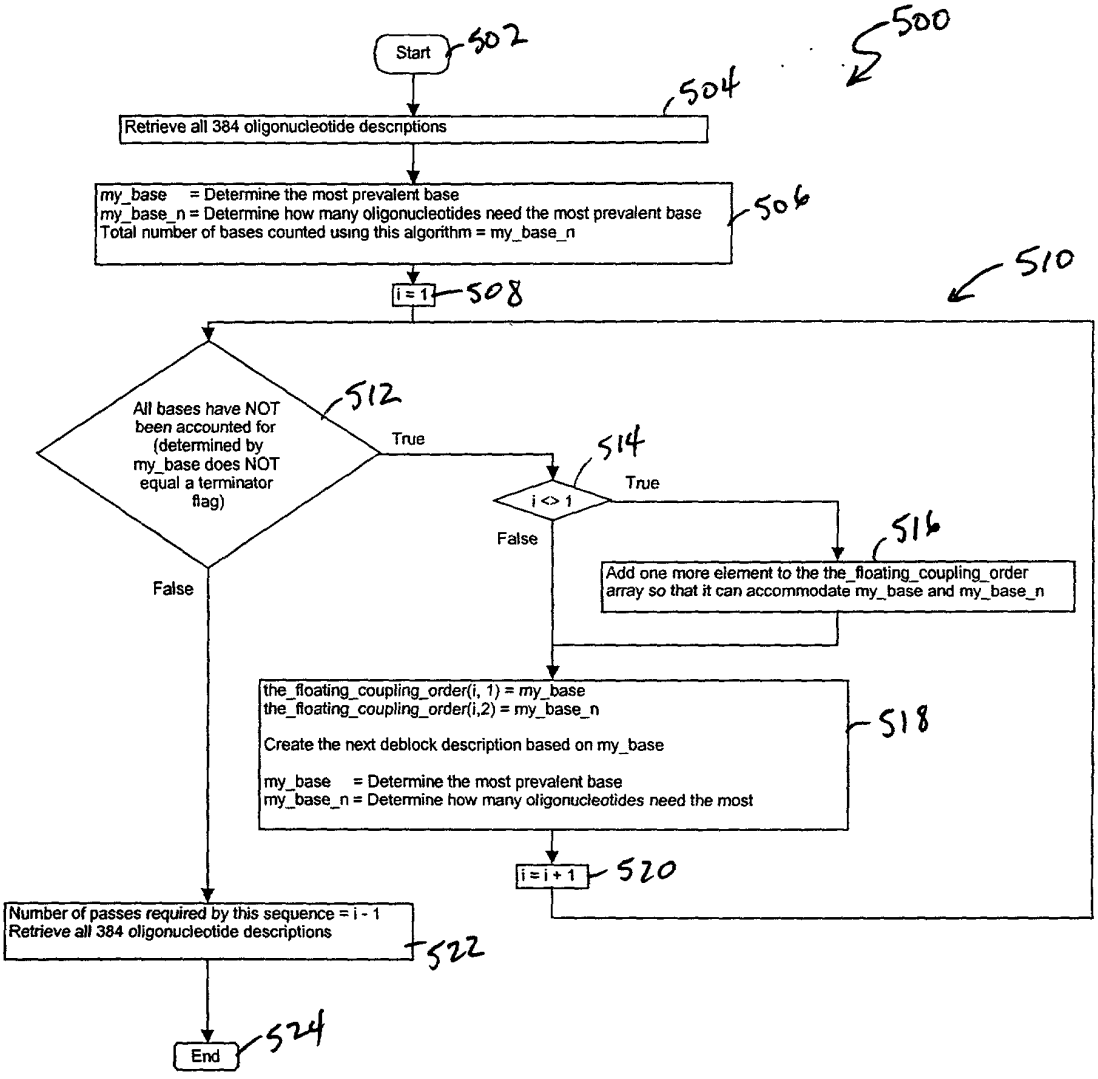

Start
502
500
504
Retrieve all 384 oligonucleotide descriptions
my_base = Determine the most prevalent base
my_base_n = Determine how many oligonucleotides need the most prevalent base
Total number of bases counted using this algorithm = my_base_n
506
i = 1
508
510
All bases have NOT been accounted for (determined by my_base does NOT equal a terminator flag)
512
True
False
514
i <> 1
True
False
516
Add one more element to the the_floating_coupling_order array so that it can accommodate my_base and my_base_n
the_floating_coupling_order(i, 1) = my_base
the_floating_coupling_order(i,2) = my_base_n
Create the next deblock description based on my_base
my_base = Determine the most prevalent base
my_base_n = Determine how many oligonucleotides need the most
518
i = i + 1
520
Number of passes required by this sequence = i - 1
Retrieve all 384 oligonucleotide descriptions
522
End
524

Figure 47

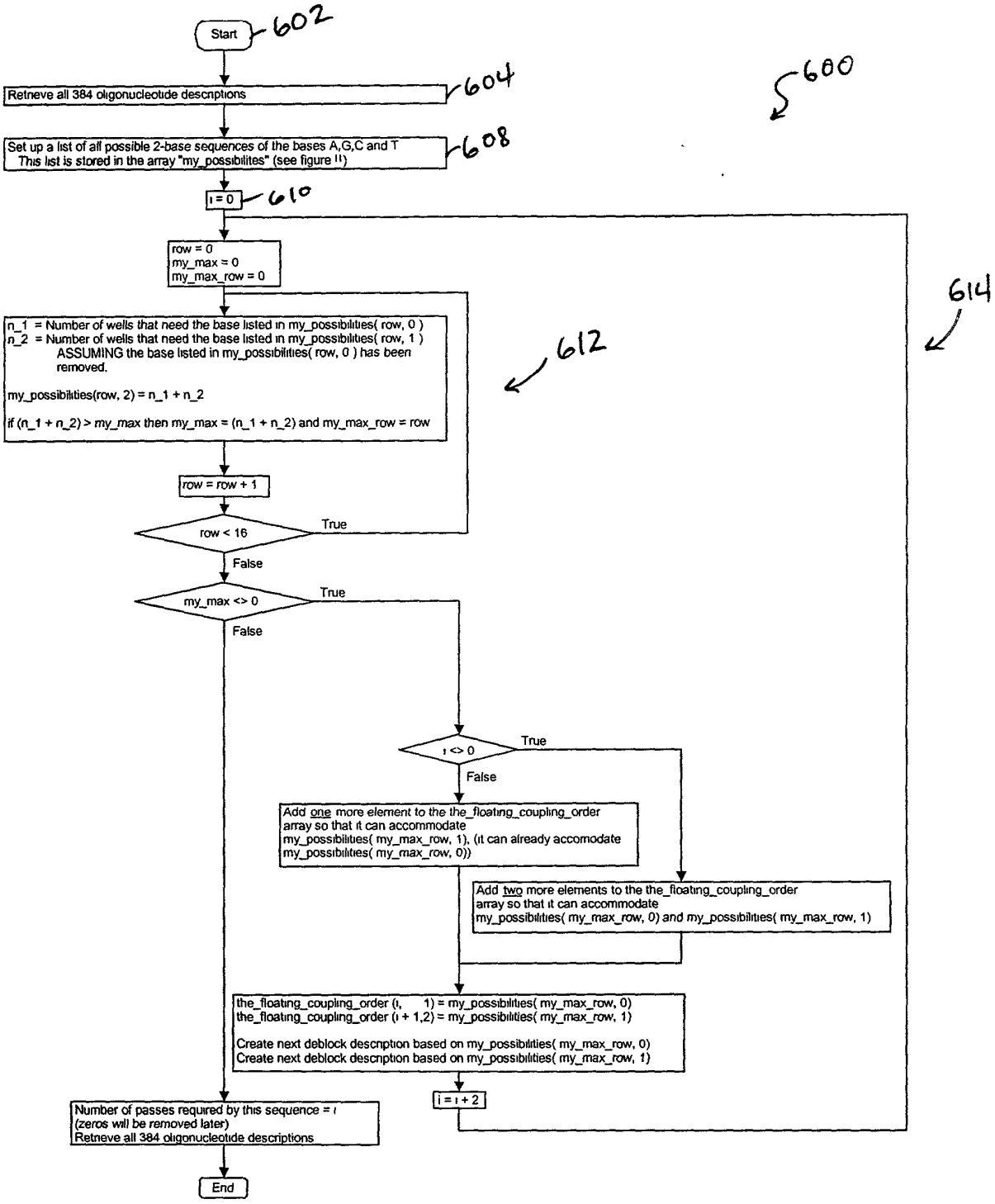
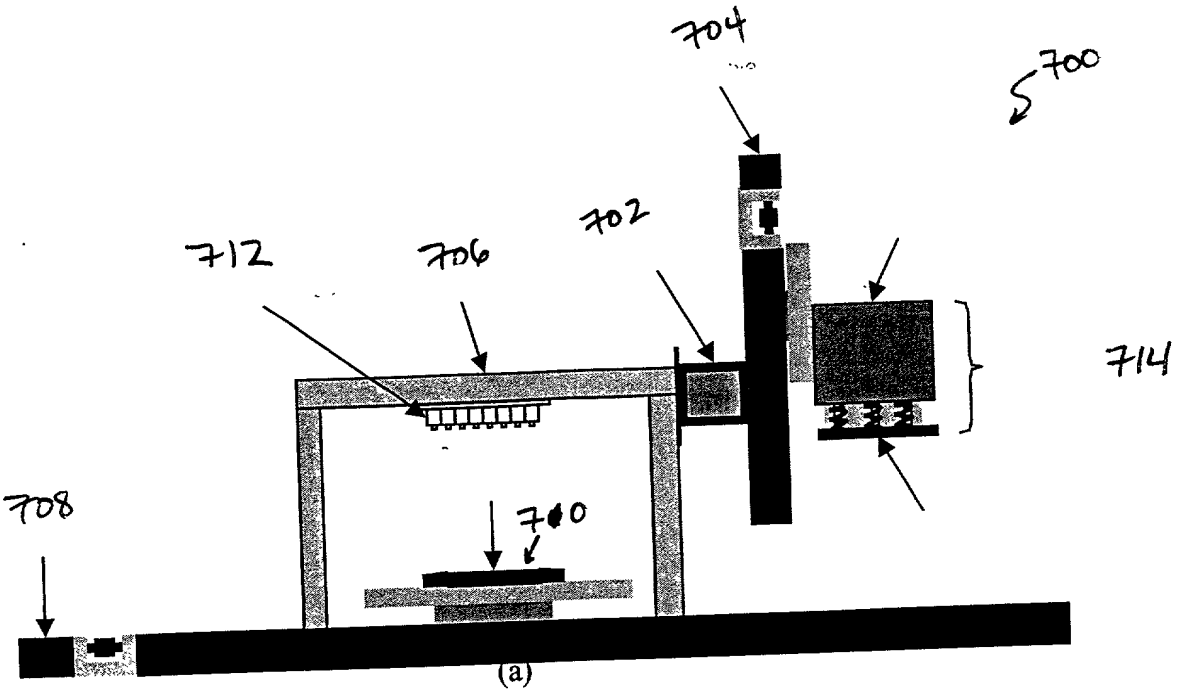
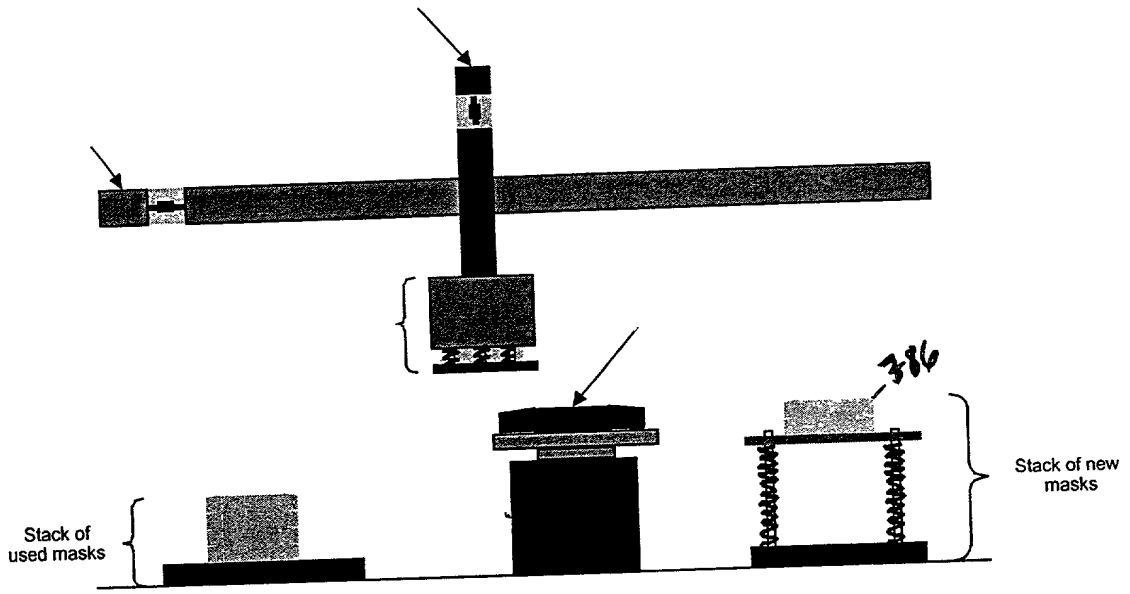

Start
602
600
Retneve all 384 oligonucleotide descnptions
604
Set up a list of all possible 2-base sequences of the bases A,G,C and T
This list is stored in the array "my_possibilites" (see figure 11)
608
i = 0
610
row = 0
my_max = 0
my_max_row = 0
614
n_1 = Number of wells that need the base listed in my_possibilites(row, 0)
n_2 = Number of wells that need the base listed in my_possibilites(row, 1)
ASSUMING the base listed in my_possibilites(row, 0) has been removed.
612
my_possibilities(row, 2) = n_1 + n_2
if (n_1 + n_2) > my_max then my_max = (n_1 + n_2) and my_max_row = row
row = row + 1
row < 16
True
False
my_max <> 0
True
False
i <> 0
True
False
Add one more element to the the_floating_coupling_order
array so that it can accommodate
my_possibilities(my_max_row, 1), (it can already accomodate
my_possibilities(my_max_row, 0))
Add two more elements to the the_floating_coupling_order
array so that it can accommodate
my_possibilities(my_max_row, 0) and my_possibilities(my_max_row, 1)
the_floating_coupling_order (i, 1) = my_possibilities(my_max_row, 0)
the_floating_coupling_order (i + 1,2) = my_possibilities(my_max_row, 1)
Create next deblock descnption based on my_possibilities(my_max_row, 0)
Create next deblock descnption based on my_possibilities(my_max_row, 1)
i = i + 2
Number of passes required by this sequence = i
(zeros will be removed later)
Retneve all 384 oligonucleotide descriptions
End

Figure 48


704
700
712
706
702
714
708
710

(a)


386
Stack of new masks
Stack of used masks

(b)

Figure 49

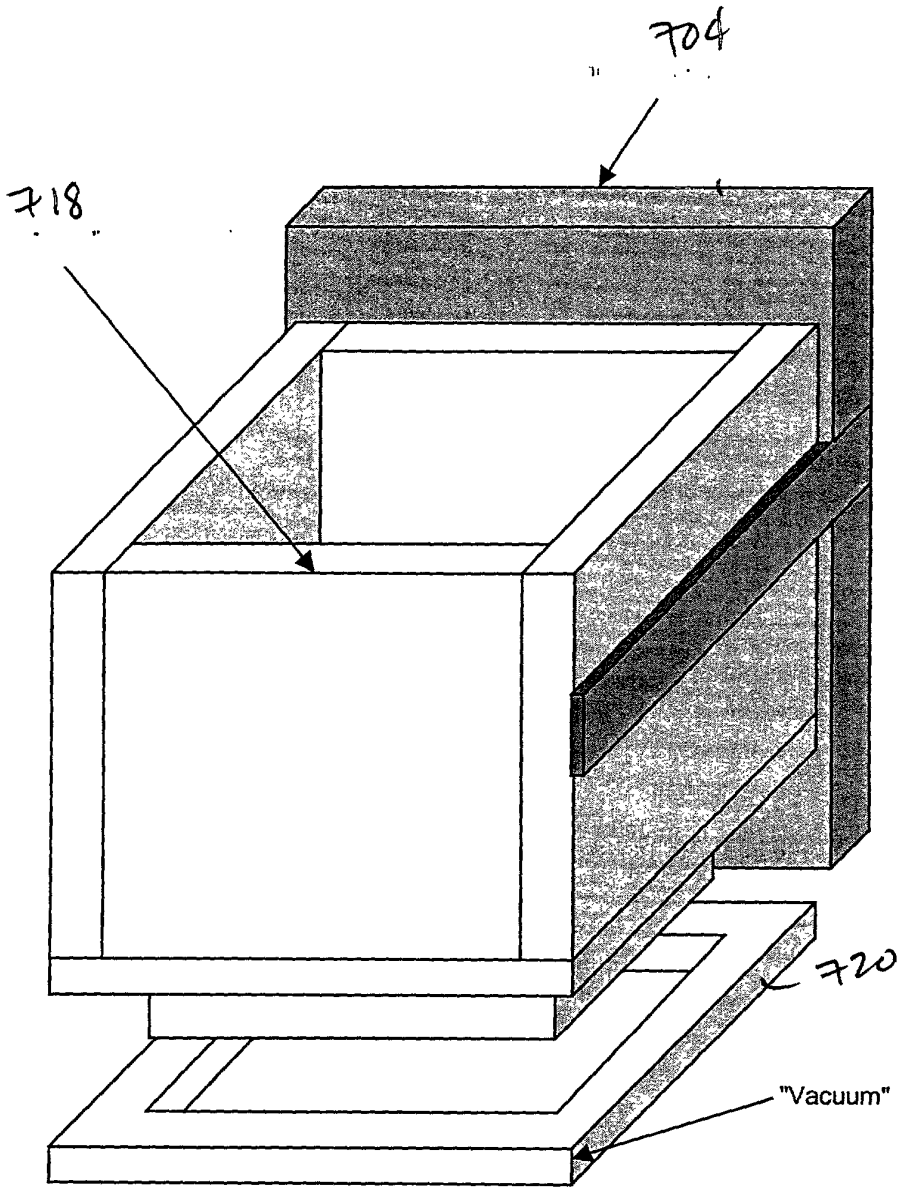


Figure 50

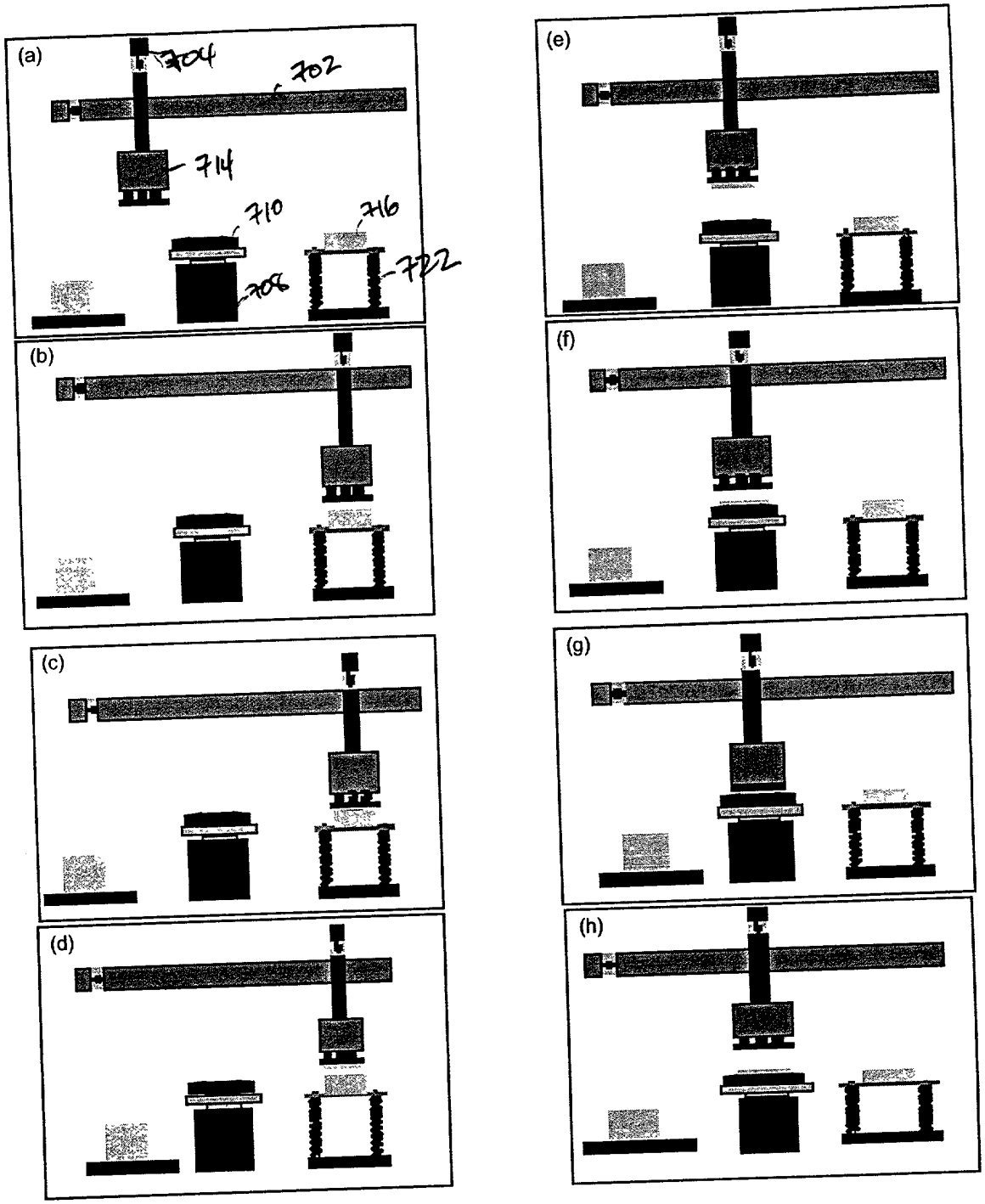


Figure 51

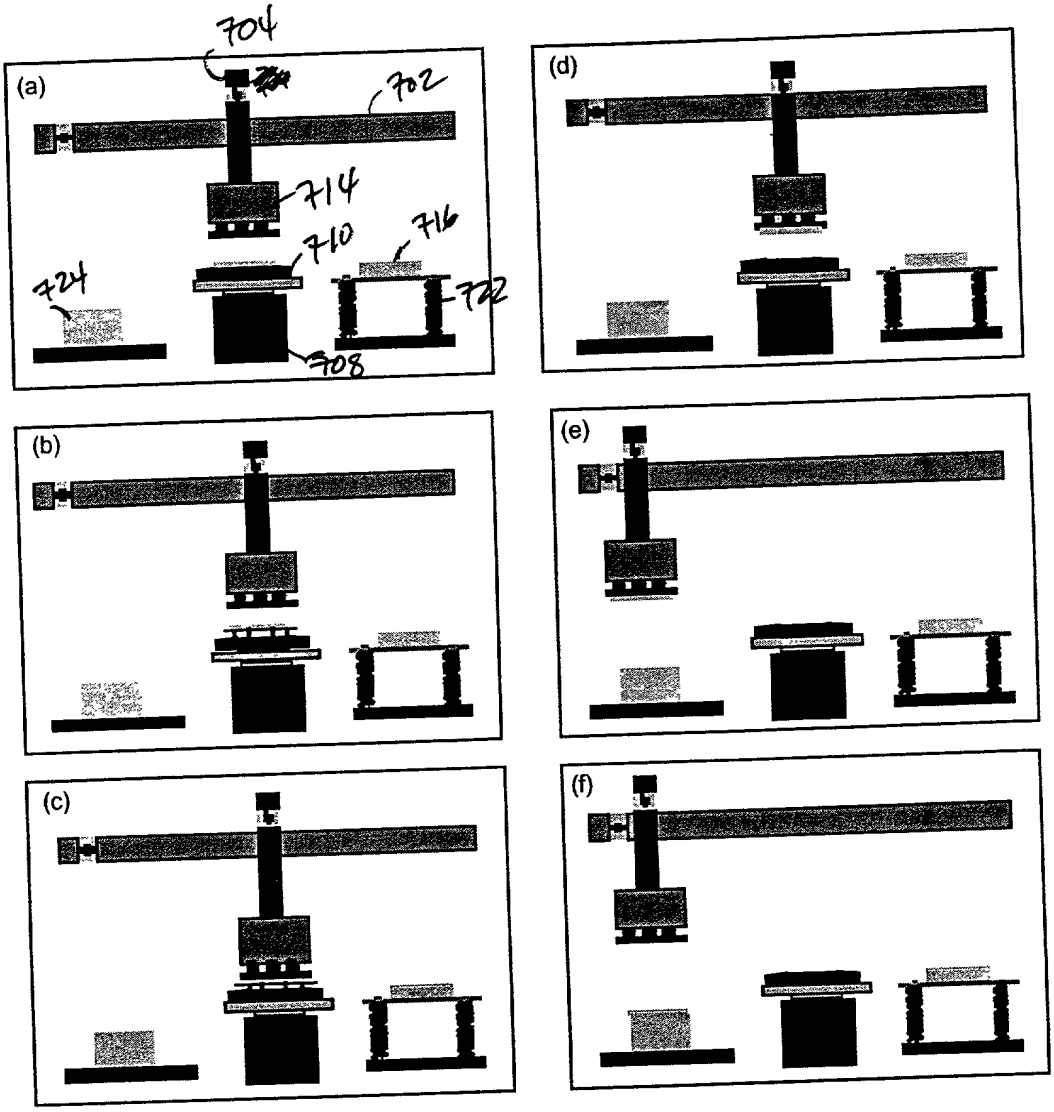


Figure 52